

AD-A063 921

COMPUTER SCIENCES CORP SILVER SPRING MD SYSTEM SCIEN--ETC F/G 1/3  
THE PREDESIGN PHASE OF THE SECOND-GENERATION COMPREHENSIVE HELI--ETC(U)  
OCT 78

DAAJ02-77-C-0057

UNCLASSIFIED

USARTL-TR-78-41

NL

1 OF 5  
AD  
A063921





19  
18 USARTL-TR-78-41

LEVEL 11



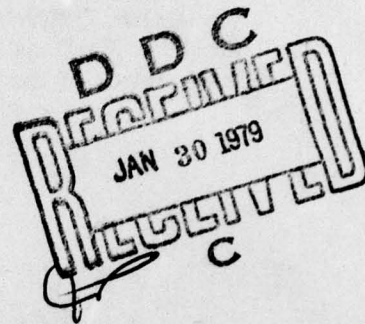
12  
S

AD A0 63921

6

THE PREDESIGN PHASE OF THE SECOND-GENERATION  
COMPREHENSIVE HELICOPTER ANALYSIS SYSTEM.

COMPUTER SCIENCES CORPORATION  
System Sciences Division  
8728 Colesville Road  
Silver Spring, Md. 20910



DDC FILE COPY

11

October 1978

12 96p.

9

Final Report, ~~submitted~~ 8 September 1977 - 22 May 1978,

15 DAAJ02-77-C-0057

Approved for public release;  
distribution unlimited.

16 1L263211D157

17 18

Prepared for  
APPLIED TECHNOLOGY LABORATORY  
U. S. ARMY RESEARCH AND TECHNOLOGY LABORATORIES (AVRADCOM)  
Fort Eustis, Va. 23604

408 479

JEB

79 01 29 018

## APPLIED TECHNOLOGY LABORATORY POSITION STATEMENT

This report summarizes the results of the Predesign phase of the Second-Generation Comprehensive Helicopter Analysis System. The predesign phase was conducted to provide: improvements to the Government-written functional specification, conceptual system design, definition of necessary computer program configuration items, development specifications, and a baseline development plan. This phase will be subsequently followed by the development, validation, maintenance, and user application phases. The report is considered to be technically sound.

Technical program direction was provided by Messrs. W. D. Vann and A. E. Ragosta, Contracting Officer's Representatives (Technical) of the Applied Technology Laboratory, Mr. H. I. MacDonald, Team Leader, and Messrs. E. E. Austin, D. J. Merkley, and P. H. Mirick of the project team.

### DISCLAIMERS

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission, to manufacture, use, or sell any patented invention that may in any way be related thereto.

Trade names cited in this report do not constitute an official endorsement or approval of the use of such commercial hardware or software.

### DISPOSITION INSTRUCTIONS

Destroy this report when no longer needed. Do not return it to the originator.



Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER USARTL-78-41 ✓	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE PREDESIGN PHASE OF THE SECOND- GENERATION COMPREHENSIVE HELICOPTER ANALYSIS SYSTEM		5. TYPE OF REPORT & PERIOD COVERED Final Report 9/8/77 to 5/22/78
7. AUTHOR(s)		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Computer Sciences Corporation ✓ System Sciences Division 8728 Colesville Road, Silver Spring, MD 20910		8. CONTRACT OR GRANT NUMBER(s) DAAJ02-77-C-0057 44
11. CONTROLLING OFFICE NAME AND ADDRESS Applied Technology Laboratory, U.S. Army Research and Technology Laboratories (AVRADCOM), Fort Eustis, VA 23604		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 63211A 1L263211D157 18 002 EK
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE October 1978
		13. NUMBER OF PAGES 395
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)  Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Prepared in cooperation with Bell Helicopter Textron P.O. Box 482 Fort Worth, TX 76101		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Acoustics, Aerodynamic Loading, Aerodynamics, Aeroelasticity, Computer Programs, Control, Digital Simulation, Dynamics, Flight Simulation, Helicopter, Instability, Loads (Forces), Mathematical Models, Noise, Performance, Rotary-Wing Aircraft, Stability		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report summarizes the results of the Predesign Phase work for the Second-Generation Comprehensive Helicopter Analysis System. The report includes an executive summary, summarizes a conceptual design for the System, describes the System's capability provided in the First Level Release and Second Level Release, discusses how the System would be used, presents a summary of the plans for development of the System, and presents dis- cussions of risk considerations. ————— over		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

79 01 29 018

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

20. (Cont'd)

The objectives and life cycle of the System are presented in terms of the six life-cycle phases: planning, predesign, development, validation, maintenance, and user applications. Predesign Phase objectives and results are discussed. An overview of the System design is provided for the engineering manager, the engineering user, and the programmer. Four major design characteristics that the System must possess for it to be universally accepted by the helicopter analysis community are discussed: user orientation, efficiency, transportability, and extendability.

The mathematical basis for the System is presented in terms of the types of problems to be solved by the System, the finite element approach, the coupling of components, aerodynamic effects, and numerical analysis considerations. The System's software architecture is presented hierarchically. The System is divided into two principal parts: the Operational Complex and the Support Complex. Each complex is made up of subsystems which in turn are made up of packages; packages are divided into subpackages. The Operational Complex contains the bulk of the System's software and solves the helicopter engineer's problems in the five technical areas of performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability. Solutions are provided at levels of complexity corresponding to the three aircraft life-cycle phases of preliminary design, detailed design, and research. The Support Complex provides the software needed to support the development, test, configuration management, and documentation of the System.

Capabilities for the First Level and Second Level Releases are presented in terms of the capabilities of each individual package or subpackage of software.

The use of the System is presented from four viewpoints: performing standard analyses, developing new analysis capabilities, using the System interactively, and using the System in a support capacity.

The plan for developing the System is presented in four major parts: organization and responsibilities, management plans, technical plans, and an implementation plan. Responsibilities for each of the Development Phase organizations are given. A dedicated project organization is recommended, and the responsibilities of the elements of that organization are defined. Management plans for development control, work management, communication, internal review and reporting, configuration management, and documentation management are presented. Technical plans include those for quality assurance, testing, and documentation. The implementation plan establishes a preliminary time-phased plan for developing the First Level and Second Level Releases of the System. The plan is presented in terms of the packages and subpackages to be provided.

The report concludes with a discussion of risk considerations in the areas of helicopter analysis and software executive overhead. For each risk consideration, alternatives that could increase or decrease risk are identified, criteria for evaluating the alternatives are presented, and the characteristics of the design which minimize each risk element are described.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)



## PREFACE

Technical project direction was provided by Messrs. William D. Vann and Arthur E. Ragosta, Contracting Officer's Representatives (Technical) of the Applied Technology Laboratory; Mr. H. I. MacDonald, Team Leader; and Messrs. E. E. Austin, D. J. Merkley, and P. H. Mirick of the project team. Principal Computer Sciences Corporation (CSC) personnel involved in the activities that resulted in this report were Messrs. Frank J. Douglas, Project Manager; Clark Oliphint, Principal System Designer; and Thomas L. Clark. Dr. P. R. Pamidi of CSC contributed to the formulation of the mathematical basis of the System. Mr. Tyce T. McLarty, the Project Engineer from Bell Helicopter Textron (BHT), managed, and was the principal contributor to, the helicopter analysis concepts included in the report. Dr. S. Eugene Sadler of BHT was the principal contributor to the aerodynamic flow field analysis.

ACCESSION for \_\_\_\_\_  
NTIS \_\_\_\_\_ Section ☒  
DOC \_\_\_\_\_ Section ☐  
UNCLASSIFIED ☐  
J.S. \_\_\_\_\_

FY \_\_\_\_\_  
DISTRICT/STATE/CITY CODES  
SPECIAL \_\_\_\_\_

A

## TABLE OF CONTENTS

Preface. . . . .	3
List of Illustrations. . . . .	8
List of Tables . . . . .	10
 <u>Section 1 - Executive Summary</u> . . . . .	 11
1.1 Objectives and Life Cycle of the System . . . . .	11
1.1.1 Planning Phase . . . . .	12
1.1.2 Predesign Phase . . . . .	12
1.1.3 Development Phase . . . . .	13
1.1.4 Validation Phase . . . . .	15
1.1.5 Maintenance Phase . . . . .	15
1.1.6 User Applications Phase . . . . .	16
1.2 Predesign Phase Objectives and Results . . . . .	16
1.2.1 Improvement of the Initial Type A System Specification . . . . .	16
1.2.2 System Capabilities . . . . .	17
1.2.3 Design of the System . . . . .	20
1.2.4 Computer Program Configuration Items . . . . .	20
1.2.5 Type B5 Development Specifications . . . . .	21
1.2.6 Baseline Development Plan . . . . .	21
1.3 Overview of the System Design . . . . .	24
1.3.1 An Engineering Manager's Overview of the System . . . . .	25
1.3.2 An Engineering User's Overview of the System . . . . .	30
1.3.3 A Programmer's Overview of the System . . . . .	33
1.3.4 A User's Overview of System Data . . . . .	36
1.4 Major System Design Considerations: Usability, Efficiency, Transportability, and Extendability . . . . .	38
1.4.1 Usability . . . . .	38
1.4.2 Efficiency . . . . .	39
1.4.3 Transportability . . . . .	40
1.4.4 Extendability . . . . .	41
 <u>Section 2 - System Design Summary</u> . . . . .	 43
2.1 Mathematical Basis for the System . . . . .	43
2.1.1 Type of Problems To Be Solved by the System . . . . .	44
2.1.2 Finite Element Concept . . . . .	50
2.1.3 Coupling of Components . . . . .	52
2.1.4 Aerodynamic Effects . . . . .	58
2.1.5 Numerical Analysis Considerations . . . . .	60
2.1.6 Potential of the System Design to Accommodate Advances in Helicopter Analysis Technology . . . . .	66
2.2 System Hierarchy . . . . .	68



## TABLE OF CONTENTS (Cont'd)

### Section 2 (Cont'd)

2.3	Operational Complex .....	71
2.3.1	Technology Component .....	76
2.3.2	Executive Component .....	101
2.4	Support Complex .....	114
2.4.1	Development Support Subsystem .....	115
2.4.2	Testing Support Subsystem .....	121
2.4.3	Configuration Management Support Subsystem .....	124
2.4.4	Documentation Support Subsystem .....	128
2.5	System Command Sequences and Data Flow .....	131
2.5.1	System Command Sequences .....	134
2.5.2	Details of Subsequences .....	143

### Section 3 - System Capability .....

3.1	First Level Release .....	156
3.1.1	First Level Release Analysis Capabilities .....	158
3.1.2	First Level Release Executive/Support Capabilities .....	170
3.2	Second Level Release .....	176
3.2.1	Second Level Release Analysis Capabilities .....	177
3.2.2	Second Level Release Executive/Support Capabilities .....	189

### Section 4 - System Use .....

4.1	System Use Overview .....	191
4.2	Master Data Base .....	195
4.3	User Input .....	200
4.3.1	Case Specification Section .....	201
4.3.2	Configuration Specification Section .....	204
4.3.3	Conditions Specification Section .....	211
4.3.4	Failure/Damage Specification Section .....	212
4.3.5	Options Specification Section .....	213
4.3.6	Accuracy Assessment Specification Section .....	218
4.4	Master Command File .....	218
4.5	System Commands .....	219
4.6	Run Data Base .....	222
4.7	Output Data .....	223
4.8	Diagnostic Messages .....	224

## TABLE OF CONTENTS (Cont'd)

<u>Section 5 - Development Plan</u> .....	226
5.1 Organization and Responsibilities .....	227
5.1.1 Development Phase Organizational Environment .....	231
5.1.2 Project Organization .....	233
5.2 Management Plans .....	246
5.2.1 Development Control Plan .....	246
5.2.2 Work Management Plan .....	249
5.2.3 Communication Plan .....	259
5.2.4 Internal Review and Reporting Plan .....	263
5.2.5 Configuration Management Plan .....	265
5.2.6 Documentation Management Plan .....	269
5.3 Technical Plans .....	270
5.3.1 Quality Assurance Plan .....	270
5.3.2 Testing Plan .....	275
5.3.3 Documentation Plan .....	289
5.3.4 Subordinate Technical Plans .....	306
5.4 Implementation Plan .....	318
5.4.1 Implementation Plan for the Operational Complex .....	324
5.4.2 Implementation Plan for the Support Complex .....	354
<u>Section 6 - Risk Considerations</u> .....	364
6.1 Helicopter Analysis Risk Considerations .....	364
6.1.1 Coupling of Components .....	364
6.1.2 Aerodynamic Flow Field Analysis .....	368
6.1.3 Numerical Integration .....	374
6.2 Executive Overhead Considerations .....	376
6.2.1 System Command Execution .....	377
6.2.2 Data Base Management Capability .....	378
6.2.3 Small Problems .....	379
6.2.4 Memory Overhead .....	380
<u>References</u> .....	382
<u>Glossary</u> .....	387

## LIST OF ILLUSTRATIONS

### Figure

1	Engineering User's View of the System . . . . .	31
2	Top-Level Hierarchy of the System . . . . .	34
3	Major Subdivisions of the Software System Hierarchy . . . . .	70
4	Hierarchy of Subsystem, Package, Subpackage, and Module . . . . .	72
5	Control and Data Flow for the Operational Complex . . . . .	74
6	Hierarchical Definition of the Technology Component . . . . .	78
7	Hierarchical Definition of the Simulation Model Initialization Subsystem. . . . .	82
8	Hierarchical Definition of the Simulation Model Subsystem . . . . .	84
9	Hierarchical Definition of the Trim Solution Subsystem. . . . .	88
10	Hierarchical Definition of the Maneuver Subsystem . . . . .	90
11	Hierarchical Definition of the Stability and Control Subsystem . . . . .	92
12	Hierarchical Definition of the Acoustics Subsystem . . . . .	94
13	Hierarchical Definition of the Aeroelastic Stability Subsystem . . . . .	96
14	Hierarchical Definition of the General Mathematical Operations Subsystem. . . . .	98
15	Hierarchical Definition of the External Models Interface Subsystem . . . . .	99
16	Hierarchical Definition of the Accuracy Assessment Subsystem . . . . .	100
17	Hierarchical Definition of the Executive Component . . . . .	102
18	Hierarchical Definition of the User Interface Subsystem . . . . .	104
19	Hierarchical Definition of the Run-Time Management Subsystem . . . . .	106
20	Hierarchical Definition of the Data Base Management Subsystem. . . . .	110
21	Hierarchical Definition of the Operating System Service Subsystem. . . . .	112
22	Hierarchical Definition of the Support Complex . . . . .	116
23	Hierarchical Definition of the Development Support Subsystem . . . . .	118
24	Hierarchical Definition of the Testing Support Subsystem . . . . .	122
25	Hierarchical Definition of the Configuration Management Support Subsystem . . . . .	126
26	Hierarchical Definition of the Documentation Support Subsystem. . . . .	129
27	Performance System Command Sequence . . . . .	135
28	Stability and Control System Command Sequence . . . . .	139
29	Loads and Vibrations System Command Sequence. . . . .	141



## LIST OF ILLUSTRATIONS (Cont'd)

### Figure

30	Acoustics System Command Sequence . . . . .	142
31	Aeroelastic Stability System Command Sequence . . . . .	144
32	Simultaneous Iterate-To-Trim Subsequence . . . . .	146
33	Simulation Model Subsequence for Preliminary Design for All Configurations . . . . .	150
34	Simulation Model Subsequence for Detailed Design of a Tandem-Rotor Helicopter . . . . .	153
35	Example of Sets in the Master Data Base . . . . .	199
36	Lines of Communication During the Development Phase . . . . .	234
37	Preliminary Development Phase Project Organization . . . . .	235
38	The Relationship of System Testing to the Software System Hierarchy . . . . .	278
39	Development and Testing Dependencies . . . . .	282
40	Milestone Schedule for Development of the Operational Complex . . . . .	325
41	Milestone Schedule for Development of a Typical CPCI of the Operational Complex . . . . .	330
42	Milestone Schedule for Development of the Support Complex . . . . .	355
43	Time-Phased Integration of the Operational Complex and Support Complex Implementation Schedules . . . . .	357

## LIST OF TABLES

### Table

1	Subsequence Sets and Functions . . . . .	133
2	Achievement of Aircraft Technical Characteristic and Life Cycle Phase Analysis Capabilities in the First and Second Level System Releases . . . . .	157
3	Allocation of the 19 Items of Paragraph (e) of Task I of the Statement of Work to the Subplans of the Development Plan . . . . .	228
4	Relationships Among the Organizations Involved in System Development and Maintenance. . . . .	244
5	System Testing Characteristics . . . . .	277
6	Involvement of the Government Project Office and Project Technical Areas in System Testing . . . . .	280
7	Test Documentation Availability . . . . .	283
8	Relationship of Testing Level to Test Tools . . . . .	285
9	Project Technical Areas Involved in Resolving Problems . . . . .	287
10	Summary of System Size and Professional Labor Estimates . . . . .	321
11	Build Sequence for Achievement of Aircraft Technical Characteristic and Life Cycle Phase Analysis Capabilities . . . . .	329
12	Build 1 of the Operational Complex . . . . .	331
13	Build 2 of the Operational Complex . . . . .	333
14	Build 3 of the Operational Complex . . . . .	337
15	Build 4 of the Operational Complex . . . . .	340
16	Build 5 of the Operational Complex . . . . .	343
17	Build 6 of the Operational Complex . . . . .	344
18	Build 7 of the Operational Complex . . . . .	348
19	Build 8 of the Operational Complex . . . . .	352
20	Build A of the Support Complex . . . . .	358
21	Build B of the Support Complex . . . . .	359
22	Build C of the Support Complex . . . . .	361
23	Build D of the Support Complex . . . . .	362

## SECTION 1 - EXECUTIVE SUMMARY

The Government and the helicopter industry need a capability to accurately predict helicopter performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability for a variety of aircraft configurations. This capability is necessary to reduce engineering development risk for new aircraft, minimize delays in deployment of new aircraft, reduce reliability and maintainability problems of operational aircraft, and prevent undue restrictions of operational capabilities of Army helicopters due to unsolved technical problems. Although the primary requirement is for accuracy, economy and reliability are important secondary requirements.

### 1.1 OBJECTIVES AND LIFE CYCLE OF THE SYSTEM

To meet the analysis needs of the helicopter community, a program, entitled the Second-Generation Comprehensive Helicopter Analysis System Program, has been established. The primary objectives of the program are (1) to develop and demonstrate a Second-Generation Comprehensive Helicopter Analysis System, which will be a major step toward satisfying the need for accurate prediction of performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability of helicopters of various sizes and rotor types, and (2) to provide the major helicopter manufacturers and Government users an operational capability using the System at their own computer facilities. Successful accomplishment of these objectives will provide an analysis capability that can subsequently evolve into a System that is more reliable and economical as well as accurate.

To satisfy the objectives of the program, a project has been established for the development of a computer-implemented Second-Generation Comprehensive Helicopter Analysis System, hereinafter referred to as the System. This System will provide a unified treatment of performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability and will be applicable to all stages in



the research, development, improvement, and employment of helicopters. Key concepts for this project include: systematic development, thorough documentation, exhaustive validation by comparison with test data, use of modern computer hardware and advanced software techniques, data management, configuration management, varying levels of complexity in the analysis techniques and representation of helicopter components, computer program modularity, user aids including diagnostics and graphics, standardized engineering notation, engineer readable program coding, development keyed to Government and industry users, and coupled aerodynamic and dynamic analysis.

The Second-Generation Comprehensive Helicopter Analysis System effort will consist of six phases: planning, predesign, development, validation, maintenance, and user applications. Each of the phases is described in the sections below.

#### **1.1.1 Planning Phase**

The specific needs for the System have been defined and an approach to be taken throughout development has been tentatively established. The initial activity of the System development effort was to define the approach to be taken throughout the development of the System. The Government/Industry Working Group (GIWG) was established and participated in an advisory capacity to formulate the overall approach to be taken. An Initial Type A System Specification was written with the advice of the GIWG detailing the functional capabilities that the System should possess. Each of the six helicopter companies represented on the GIWG also provided the Army with comments on the technical approach.

#### **1.1.2 Predesign Phase**

This final report documents the work accomplished by Computer Sciences Corporation (CSC) and its subcontractor, Bell Helicopter Textron (BHT), during the Predesign Phase. The CSC/BHT team was one of three teams selected to improve the Initial Type A System Specification; define the feasible First Level Release, Second Level Release, and Long Range System capabilities; provide a top-level design

for the System; define the Computer Program Configuration Items (CPCIs) that compose the System; produce a set of Type B5 Development Specifications; and produce a Baseline Development Plan. The Government project team has been advised by the GIWG to enhance user orientation and by a Technical Advisory Group (TAG) to enhance the technical approach. The Government will review results of this phase, prepare a revised Type A System Specification, and formulate tentative requirements for experimental data to determine CPCI and System accuracy.

#### **1.1.3 Development Phase**

During this phase, the First Level Release and Second Level Release capabilities will be developed in accordance with the Type A System Specification defined in the Predesign Phase and in general accordance with AMCP 70-4, Research and Development Software Acquisition - A Guide for the Material Developer. The First Level Release of the System will be developed using state-of-the-art rotary-wing technology and software techniques as extensively as possible without undue sacrifice in the potential of the Second Level Release and Long Range System capabilities. The Second Level Release of the System will be developed using more advanced rotary-wing technology and software techniques than used for the First Level Release. It will incorporate corrections for errors and deficiencies which will have been identified after the First Level Release, as well as additional functional capabilities not developed in the First Level Release.

The Development Phase contractor, expected to be one of the Predesign Phase contractors, will be responsible for

- Designing the System
- Identifying CPCIs
- Preparing a Type B5 Development Specification for each CPCI, for both First Level Release and Second Level Release capabilities



- Recommending those CPCIs to be developed by the Development Phase Contractor, those by subcontractors, and those to be Government-furnished based on the premises that few, if any, First Level Release CPCIs will be Government-furnished and few, if any, Second Level Release CPCIs will be developed by subcontractors
- Developing those CPCIs approved by the Contracting Officer
- Determining that each CPI meets the requirements and quality assurance provisions of its Type B5 Development Specification
- Integrating all CPCIs into the System
- Conducting a functional demonstration of the System to demonstrate to Government and industry that the System meets the requirements and quality assurance provisions of the Type A System Specification
- Defining a unified documentation approach and editing documentation for each CPI to promote uniformly high standards
- Implementing a configuration management plan
- Providing training and maintenance support to Government and industry users during the initial portion of the Validation Phase

The GIWG and the TAG will continue to advise the Government project team during the Development Phase.

The Government will monitor the development of the System in detail down to the level of a single line of code or engineering equation. The Government will approve the Type B5 Development Specifications produced by the Development Phase Contractor for each CPI. The Government will, in addition, exercise selection approval of subcontractors for CPI development. The Government will prepare to assume full responsibility for the System during the Maintenance Phase. The

Government will finalize requirements for, and sponsor acquisition of, experimental data necessary to determine CPCI and System accuracy.

#### **1.1.4 Validation Phase**

The objectives of the Validation Phase are to establish within the Government/industry user community an operational capability with the System, contribute to the validation of the accuracy and operating cost of the System, and provide inputs from the user community to the Development Phase Contractor and the Government project team to maximize user orientation of the System during the Development and Maintenance Phases.

Helicopter manufacturers under contract to the Government will validate the applicability of the System to their helicopter types through correlation with experimental data; these contracts will be separate from the Development Phase contract. The helicopter manufacturers, along with Government users, will

- Achieve an operational capability with the System
- Apply the System to current rotary-wing research and development efforts, in parallel with other methods of analysis, to evaluate the effectiveness of the System
- Identify minor errors and deficiencies, determine corrective measures, and recommend their implementation
- Recommend System enhancements to the Government project team

#### **1.1.5 Maintenance Phase**

The Maintenance Phase will be a continuous activity consisting of System correction, modification, and development in response to errors and deficiencies identified by the user community. Further advancements in the state-of-the-art in rotary-wing analysis and computer technology will also be incorporated. The

responsibility for maintenance will be assumed by the Applied Technology Laboratory, which will serve as the focal point for dissemination of documentation and advice on operational problems encountered using the System.

#### **1.1.6 User Applications Phase**

At the beginning of this phase, the Government/industry user community will have attained a mature operational capability with the System. With their own funds, users will utilize the System capabilities for their own analysis needs. They will continue to provide the Government with input to the maintenance activity so that the System will continue to meet their needs.

### **1.2 PREDESIGN PHASE OBJECTIVES AND RESULTS**

The objectives of the Predesign Phase were to improve the Initial Type A System Specification; define the feasible First Level Release, Second Level Release, and Long Range System capabilities; produce a preliminary System design; define Computer Program Configuration Items (CPCIs) which make up the System; produce an associated set of Type B5 Development Specifications; and produce a Baseline Development Plan. The results of the contractual efforts are documented in detail in the Contract Data Items (deliverables) of the contract and are summarized in this report. The paragraphs below correspond to each objective of the Predesign Phase and present the principal results associated with meeting the objective.

#### **1.2.1 Improvement of the Initial Type A System Specification**

A primary objective of the contract was to improve the Initial Type A System Specification with special emphasis on the 20 critical issues identified in the contract. Principal recommended improvements to the Initial Type A System Specification are as follows:

- Add the major functional capability of accuracy assessment to provide an objective measure of System accuracy



- Make transportability an explicit requirement of the System so that both industry and Government users will have immediate access to the System
- Specify ANSI FORTRAN as the implementation language to facilitate System transportability and acceptance
- Clarify and extend the restart capability to ensure cost savings in day-to-day operations for large problems
- Provide an interactive tutorial capability to enhance user acceptance of the System
- Specify detailed programming standards to increase System reliability and to decrease documentation and maintenance costs
- Specify that the System design take into account virtual memory processing, hardware vector processing, parallel processing, and cache memory hardware characteristics
- Replace the original Section 4, Quality Assurance Provisions, with a more comprehensive one to ensure that the System is adequately tested
- Provide an extensive list of Particular Functional Capabilities including memory, execution-time, and cost estimates
- Specify that the computers for both the First Level Release and the Second Level Release be IBM S/370s and S/360s and CDC 6600s and CYBERs to ensure that both industry and Government users of the System will use the System at their facility

#### **1.2.2 System Capabilities**

System capabilities were allocated to the First Level Release and the Second Level Release based on (1) the prescribed budget is 20 professional man-years per year



for 4 years; (2) the schedule for completion of the First Level Release is 2 years after Development Phase contract award and the schedule for completion of the Second Level Release is 4 years after Development Phase contract award; (3) relatively high priorities are assigned to System capabilities for problem analyses for the detailed design and preliminary design aircraft life cycle phases compared to priorities for the research aircraft life cycle phase, and (4) the man-month estimates for CPCI development that were made during the Predesign Phase.

#### 1.2.2.1 First Level Release Capability

The First Level Release consists of helicopter analysis capabilities for performance, stability and control, aeroelastic stability, and rotor loads and vibrations for the aircraft life cycle phases of preliminary design and detailed design. An acoustics analysis capability is also provided for preliminary design.

The First Level Release capability in terms of physical components is as follows. The rotor representations include semiempirical equations, rigid-blade equations, and dynamic analyses for all rotor types with lag dampers, flapping stops, and lag stops. A general rigid control system is represented. The drive system representation includes rigid and static elastic analyses with an engine performance table. The airframe representation includes a rigid fuselage, aerodynamic surfaces, stores, and pylons, as well as a simple landing gear. The capability for the airmass includes steady aerodynamic coefficients using tables; unsteady aerodynamic coefficients using Theodorsen/Loewy or  $\alpha$ , A, B methods; momentum theory flow field with or without time delay; and a prescribed rotor wake. Prescribed motions of a ground or deck surface are included as well.

Executive software in a batch mode will by and large be complete for the First Level Release. The only principal Executive software capability postponed to the Second Level Release is the capability to run the System in an interactive mode. Support software is complete by the First Level Release. The First Level Release

of the System will be available on the IBM S/370 and S/360 computers and on the CDC 6600 series and CYBER series computers.

#### 1.2.2.2 Second Level Release Capability

In addition to all of the capabilities of the First Level Release, the Second Level Release provides analysis capabilities for performance, stability and control, loads and vibrations, acoustics, and aeroelastic capability for the research aircraft life cycle phase. Also included in the Second Level Release are (1) the acoustics capability for detailed design and (2) the loads and vibrations capability for the airframe, the engine/drive system, and the control system/pilot for preliminary design and detailed design. Another significant capability provided by the Second Level Release is a capability to assess the effects of damage to or failure of the various aircraft components (e.g., rotor blade, engine/drive system components). The user will be able to use all the Second Level Release analysis capabilities in an interactive mode on the IBM S/370 and S/360 and on the CDC 6600 and CYBER series computers.

The Second Level Release capability, in terms of physical components, is as follows. The rotor representations include elastic rotor blades, semiempirical circulation control rotors, semiempirical reaction drive rotors, pendulum absorbers, control load reduction devices, and servo flaps. The control system/pilot representations include elastic control systems, dynamic control systems, force feel systems, automatic flight control systems, control feedback from force/motion sensors, and pilot transfer functions. The engine/drive system representations include detailed engine analysis with a governor and fuel control devices; a rigid, static elastic, and dynamic gearbox; a dynamic driveshaft; and a clutch. The airframe representations include a static elastic and dynamic fuselage, a static elastic and dynamic aerodynamic surface, vibration control devices, suspended cargo, a complex landing gear, dynamic stores, and hoist and load stabilization



devices. The airmass representation includes free rotor wake, cable aerodynamics, unsteady aerodynamic loadings, wind tunnel wall and blockage effects, and aerodynamic interference effects between and among rotors, aerodynamic surfaces, and bodies. Also included is an aerodynamic analysis for arbitrary bodies and nonrotating lifting surfaces. Other analysis components include a dynamic test stand, an elastic or plastic deformable ground or deck surface, and a water surface.

### 1.2.3 Design of the System

The System design provides the flexibility to analyze helicopter components (e.g., rotor, fuselage) separately or in combination. Although the System design ensures accurate solutions to large problems, which characterize the research aircraft life cycle phase, the System design also provides an efficient solution to small problems, which characterize the preliminary design aircraft life cycle phase. The usability of the System has been a prime design goal throughout the contract. Engineering users will use the System as an analysis tool as an integral part of their day-to-day work. Fulfilling the needs of the engineering user while at the same time making the System easy to use (i.e., not requiring the engineering user to know the internal design of the System) has been a principal design objective. In addition, the needs of the methods developer, who will use the System as a foundation to explore new and improved analysis techniques that will eventually be incorporated in the System, have also been met. A summary of the System design is presented in Section 1.3. Section 2 presents a detailed summary of the design presented in the Predesign Phase Type B5 Development Specifications for the Second Generation Comprehensive Helicopter Analysis System, CSC/SD-78/6083.

### 1.2.4 Computer Program Configuration Items

CSC defined as Computer Program Configuration Items (CPCIs) 204 software elements of the System. This large number of software elements were designated as

CPCIs to provide the visibility necessary for determining which of the elements are candidates for development by the Government and subcontractors. However, CSC recommends that the number of CPCIs for the Development Phase be greatly reduced in order to minimize the costs associated with the procurement, documentation, and configuration management of CPCIs.

#### 1.2.5 Type B5 Development Specifications

The Type B5 Development Specifications produced during the Predesign Phase present a unified treatment of the mathematical basis of the System in which all System analyses are derived from one master set of differential equations in matrix form. Another feature of the Type B5 Development Specifications is an extensive discussion and numerous examples of how the System is used by the engineering user and the methods developer. The software design of the System is presented in a format called Hierarchy Plus Input Process Output (HIPO) that presents the elements of the design in an organized, reader-oriented fashion. Data flow diagrams show how the software elements of the System are related to each other and, most important, illustrate the flow of data within the System for the five major technical characteristics: performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability. Finally, a functional description of each CPI is presented.

#### 1.2.6 Baseline Development Plan

The System development plan, which is summarized in Section 5 of this report, consists of the following: organization and responsibilities, technical plans, management plans, and a time-phased plan for implementing the software elements of the First Level Release and the Second Level Release.

##### 1.2.6.1 Organization and Responsibilities

The plan for organization and responsibilities in the Development Phase emphasizes the need for effective communication throughout development and defines in some



detail the responsibilities of participating organizations--the Applied Technology Laboratory, the Development Phase contractor, the integrated team member subcontractor, CPCI subcontractors, the Government/Industry User Community, the Government/Industry Working Group, and the Technical Advisory Group.

#### 1.2.6.2 Technical Plans

Technical plans include those for quality assurance, testing, documentation, providing an effective software development computing environment, managing System software and data, installing the System at user sites, training, and maintenance.

The quality assurance plan features effective techniques to ensure that users receive a high-quality System: top-down development; use of modern software system design presentation techniques (e.g., Program Design Language, HIPO, data flow diagrams); structured walkthroughs for design and coding review; structured programming techniques; automated tools to assess conformance of source code to programming standards; independent testing; and a separate organizational element to ensure that these techniques are implemented.

Features of the remaining technical plans are as follows:

<u>Plan</u>	<u>Highlight</u>	<u>Benefit</u>
Testing	Test documentation plan	Provides sufficient yet cost-effective control of testing
	Automated test tools	Provides a quantitative assessment of the scope of testing
Documentation	Integration of MIL-STD-490 standards with those of DoD Manual 4120.17-M	Avoids duplication of effort, thus reducing costs
Software Development Computing Environment	Terminals dedicated to System development	Increases programmer productivity

<u>Plan</u>	<u>Highlight</u>	<u>Benefit</u>
Data Management	Effective library controls	Avoids loss in productivity in case of inadvertent destruction of tapes/disks
Installation and Release	System installation at each user's site	Enhances user acceptance
Training	Programmer training in addition to user training	Permits an installation to make local changes independent of those supplied centrally by the Applied Technology Laboratory
Maintenance	Effective communication of System status	Enhances user acceptance

#### 1.2.6.3 Management and Implementation Plans

The management plans emphasize the need for: a System Development Plan to provide a public statement of all management and technical project plans; effective communication among all organizations; internal as well as external review and reporting procedures; and configuration management controls.

The principal characteristic of CSC's time-phased plan for implementing the software elements of the First Level Release and the Second Level Release is that it has been shaped by the strategy of builds, a powerful, proven software implementation strategy that minimizes risk. Because of the System's size and complexity, it should be developed and tested incrementally. A build, which is a subset of the entire System, provides a demonstrable (i.e., testable) functional capability that is a subset of the total functional capability of the System. The System is constructed in a sequence of builds, where each build contains all of the capability of the previous build in the sequence plus new capability. This "build-a-little, test-a-little" philosophy has several advantages over the alternative strategy of developing all the software elements required to produce the First Level Release



capability and then all the software elements required to produce the Second Level Release capability. These advantages are as follows:

- If there are any major interface problems (e.g., between Executive and Technology software), they will be discovered early enough so that corrections can be made without affecting the schedule for completing either the First Level Release or the Second Level Release.
- Integration, the phase in the development life cycle where software interface problems historically have surfaced, is spread more smoothly over the entire Development Phase rather than being placed near the end. Risk is thus reduced.
- A stable and well-defined partial System is available for testing following the first build.
- A part of the total System capability is demonstrated to the Government and to System users early so that user experience can influence the final delivered System.

### 1.3 OVERVIEW OF THE SYSTEM DESIGN

The summary of the System design in the subsections below is presented from three different points of view: that of an engineering manager (Section 1.3.1), that of an engineering user (Section 1.3.2), and that of a programmer (Section 1.3.3). An overview of the data that the System will process is allocated an entire subsection, Section 1.3.4. A separate subsection is used to demonstrate the importance that CSC attaches to a systems view of data: without such a systems approach, overall life cycle costs can escalate.

Acceptance of the System as a standard throughout the helicopter industry requires that the System possess seven characteristics:

1. Accuracy, i.e., the System must provide accurate methods to analyze helicopter configurations.

2. Usability, i.e., the System must be easy to use by the engineering analyst.
3. Transportability, i.e., it must be easy to move the System from one computer family to another, with minimum modifications.
4. Extendability, i.e., it must be easy to add new analysis capabilities and to modify existing analysis capabilities without changing software unrelated to the capability being added or modified. In addition, it must be easy to experiment with new or modified capabilities.
5. Reliability, i.e., the System must provide the engineering analyst with confidence in the adequacy of the analysis upon which the System design is based.
6. Maintainability, i.e., it must be easy to isolate and correct deficiencies in the System.
7. Efficiency, i.e., the System must be able to efficiently analyze both small and large problems.

The System characteristics of accuracy, reliability, and usability are dependent upon a mathematical basis which inherently provides a foundation for achieving these three goals and upon a design which recognizes the importance of these goals. Section 1.3.1 discusses the relationship between the mathematical basis of the system and the System characteristics of accuracy, reliability, and usability.

#### **1.3.1 An Engineering Manager's Overview of the System**

For the System to be accepted as a standard by the helicopter analysis community, it must be based on a mathematical approach which inherently provides a basis for an accurate, reliable, and user-oriented analysis capability. CSC/BHT has therefore selected a design approach that provides the analysis capability needed to

accurately evaluate helicopter designs, the reliability needed to ensure confidence in the analysis performed, and the ease of utilization needed to ensure acceptance by the individual helicopter engineer. The mathematical foundation of the System design is based on a unified mathematical approach for all analysis performed by the System and on the definition of a user-interface environment oriented to the engineer and the problem to be solved rather than to the programmer and the software solution to the problem.

#### 1.3.1.1 Summary of the Unified Mathematical Approach

The unified mathematical approach involves three basic concepts. First, helicopter analysis and simulation, in the most general form, can be represented by a system of differential equations in the form of the following matrix equation:

$$[M] \{\ddot{q}\} + [C] \{\dot{q}\} + [K] \{q\} = \{F\} \quad (1)$$

where  $\{q\}$ ,  $\{\dot{q}\}$ , and  $\{\ddot{q}\}$  are generalized displacement, velocity, and acceleration vectors, respectively;  $[M]$ ,  $[C]$ , and  $[K]$  are the mass, damping, and stiffness matrices, respectively, of the aircraft configuration under analysis (these matrices may be constant, periodic, otherwise time variant, or weak functions of the  $\{q\}$  and  $\{\dot{q}\}$  vectors); and  $\{F\}$  is a vector of generalized forces that may be constant but is more commonly periodic or otherwise time variant. In addition,  $\{F\}$  may be a linear or nonlinear function of the  $\{q\}$  and  $\{\dot{q}\}$  vectors.

All helicopter analysis problems to be analyzed by the System can be represented by this equation or variations thereof. The determination of steady-state or trim conditions, a basic requirement for virtually all helicopter analysis problems, is based on a variation of this basic equation. The natural frequencies and mode shapes of the aircraft structure and its components are derived using a variation



of this basic equation. For stability analysis, the calculation of stability derivatives is based on this same basic equation. Finally, transient aircraft maneuver problems are represented using this basic equation. The use of this single basic differential equation is the first concept that unifies the mathematical basis of the System.

The second concept that unifies the mathematical basis of the System is the finite element approach for representing helicopter components. The finite element approach has been successfully employed in structural analysis applications for many years. In recent years, the feasibility, suitability, and adaptability of this approach for analyzing helicopter-related structures have been widely demonstrated. The finite element approach provides a unified concept applicable to static analysis, dynamic analysis, and the analysis of aerodynamic effects.

The third unifying mathematical concept included in the System is the capability to analyze a physical configuration composed of independently defined components.

The finite element approach is directly applicable to combining aircraft and other components to form a variety of configurations. The CSC/BHT design provides a general and systematic method of coupling various components to yield complex structures or configurations. Two approaches to coupling of components were considered: the substructure analysis approach and the component modes approach. The CSC/BHT system design, because it is based on the finite element approach, can accommodate either approach to coupling of components. However, the projected funds available for the Development Phase may not permit both approaches to be implemented. Because of the potential computer cost savings that can be realized if the component modes approach is used for coupling of components, the component modes approach is included in the design for the First Level Release of the System. In addition, it is recommended that, if funding permits, the substructure analysis approach to component coupling be added to the Second Level Release of the System.

The finite element approach is also applicable to analyzing the aerodynamic flow field and its effect upon various components of a helicopter configuration. Aerodynamic models (rather than structural models) can be employed to define the interaction between the structure and the surrounding air mass. For maximum flexibility, the design permits aerodynamic node points to differ from dynamic node points.

The combination of a single basic differential equation, the finite element approach, and a systematic method to couple components results in a unified mathematical basis for accurately and reliably analyzing helicopter configurations.

This approach also enhances the usability of the System by the engineer because it provides a consistent and unified way of defining helicopter models, analyzing helicopter configurations, and assessing the results of the analysis. The use of specialized analytical formulations was rejected because such an approach would have a severe impact on the ease of using the System.

#### 1.3.1.2 Summary of the User's Interface

A major design goal established by CSC/BHT was to facilitate the use of the System by an engineer. Basing the System design on a unified mathematical concept is the first step in accomplishing this goal. To fully meet this goal, an engineering-oriented interface is defined that allows the engineer to specify the problem to be solved in terms of the problem components rather than in terms of the software components. This user/problem orientation eliminates the user's need to know the internal design of the System; users can thus focus their attention where it belongs: on the helicopter rather than on the System.

Six of the key elements of the user interface provided in the design are verification of input before performing the analysis; meaningful diagnostic messages that use engineering terminology rather than software terminology; capability to define input at an interactive terminal; capability to view System output at an interactive

terminal; engineering-oriented analysis reports; and graphic presentation of analysis results.

Two unique approaches included in the design eliminate the need for the engineer to understand the software design in describing either the problem to be solved or the analysis to be performed. These two key approaches are simplification of the definition of the physical configuration and simplification of the analysis definition.

One of the biggest user-interface problems associated with finite element approaches is the large amount of data required to represent a complex model. To solve this problem, the design provides an environment in which the engineer defines the problem to be solved in terms of the aircraft components comprised in the physical configuration. A data base of aircraft component descriptions will be available so that the engineer need not explicitly include a description of each component comprised in the overall physical configuration to be analyzed. To describe the physical configuration the engineer need only indicate the aircraft components to be used and the way in which they are to be coupled. If desired, modifications to the component descriptions as stored in the data base can be specified at the time the analysis run is made. If a new component is required, it either can be explicitly described in the analysis run or can first be added to the data base of components. The data base of components eliminates the need to include descriptions of frequently used and stable components in each analysis run, thus minimizing the amount of data that the engineer must supply for an analysis.

The definition of the analysis to be performed is another user problem addressed by the design. All too frequently, automated analysis systems require that the engineer understand the software design of the system in order to define the analysis to be performed. The CSC/BHT design solves this problem. In defining the analysis to be performed, the engineer need only specify the major analytical



functions to be performed (e.g., find a steady-state flight condition, calculate stability and control data, determine acoustic responses). The System itself then translates these analysis specifications into a detailed command sequence that is oriented to the software structure of the System. This approach eliminates the need for the engineer to understand the software design, thus allowing the analysis to be defined in terms of the types of analysis to be performed rather than in terms of the software elements to be executed. If desired, engineering users and developers of new algorithms and new capabilities to be incorporated in the System (these latter users are called methods developers) may construct their own detailed command sequences or modify existing ones. The detailed command sequences are retained on a data base so that new major functions or modifications to existing functions can be permanently defined for subsequent use by the engineering community at an installation. The definition of a new command sequence or a modification to an existing command sequence does not require any software modifications unless interfaces among software elements are changed.

### 1.3.2 An Engineering User's Overview of the System

The Second Generation Comprehensive Helicopter Analysis System performs the previously described helicopter analysis in three phases: input, processing, and output. A diagram of the System data flow is provided in Figure 1.

#### 1.3.2.1 Input Phase

In the input phase, the System reads a user input file, which may be either a card deck, a card-image file, or a file prepared from an interactive terminal. Information in the user input file is the basis for locating in the Master Data Base the descriptions of the aircraft configuration to be analyzed and the conditions for the analysis and placing those descriptions in the Run Data Base. Information in the user input file is also the basis for selecting, from the Master Command File, predefined sequences of System Commands which identify the sequence of operations required to perform the analysis, and placing the complete sequence in the

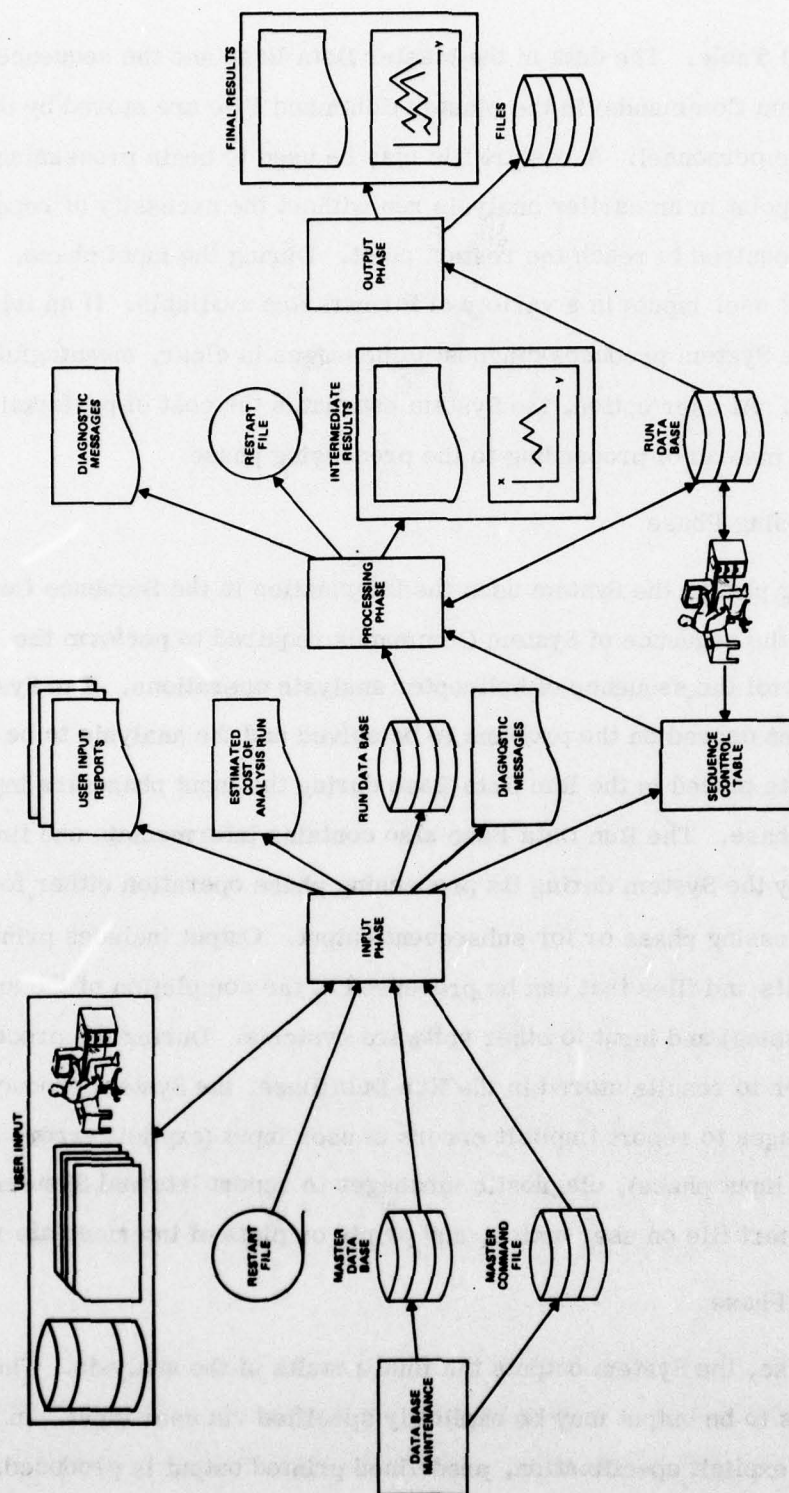


Figure 1. Engineering User's View of the System

Sequence Control Table. The data in the Master Data Base and the sequences of operations (System Commands) in the Master Command File are stored by data base maintenance personnel. A restart file may be used to begin processing from an intermediate point in an earlier analysis run without the necessity of repeating the processing required to reach the restart point. During the input phase, reports showing all user inputs in a variety of formats are available. If an input error exists, the System produces diagnostic messages in clear, meaningful, engineering terms. At user option, the System estimates the cost of performing the analysis run instead of proceeding to the processing phase.

#### 1.3.2.2 Processing Phase

In the processing phase, the System uses the information in the Sequence Control Table, which is the sequence of System Commands required to perform the analysis, to control the sequence of helicopter analysis operations. The System Commands chosen depend on the problem to be solved and the analysis to be performed. The data placed in the Run Data Base during the input phase are input to the processing phase. The Run Data Base also contains intermediate and final results produced by the System during its processing phase operation either for use later in the processing phase or for subsequent output. Output includes printed and plotted results and files that can be processed at the completion of the analysis run (post-processing) and input to other software systems. During the processing phase, in addition to results stored in the Run Data Base, the System produces diagnostic messages to report implicit errors in user input (explicit errors are diagnosed in the input phase), diagnostic messages to report internal System problems, a restart file on user option, and prints or plots of intermediate results.

#### 1.3.2.3 Output Phase

In the output phase, the System outputs the final results of the analysis. The particular quantities to be output may be explicitly specified via user input. In the absence of such explicit specification, predefined printed output is produced. The

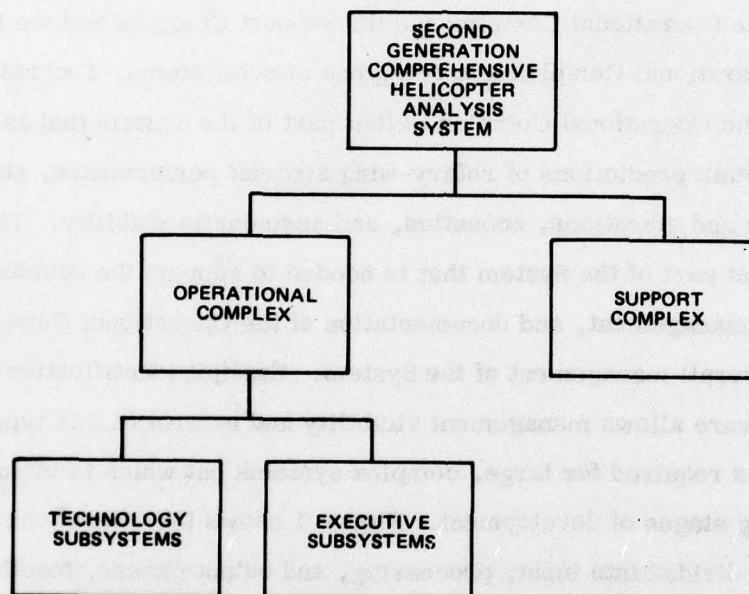


results to be output are retrieved from the Run Data Base, where they were stored during the processing phase. The user may also specify quantities output to files intended as input to programs external to the System (i.e., the External Models Functional Capability).

### 1.3.3 A Programmer's Overview of the System

The hierarchy diagram in Figure 2 provides a System overview to programmers interested in the architecture of the System. The figure shows the division of the System into the Operational Complex and the Support Complex and the further division of the Operational Complex into two types of subsystems, Technology and Executive. The Operational Complex is that part of the System that is used by the engineer to obtain predictions of rotary-wing aircraft performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability. The Support Complex is that part of the System that is needed to support the development, test, configuration management, and documentation of the Operational Complex and to support the overall management of the System. Explicit identification of Support Complex software allows management visibility and control of this type of software, which is required for large, complex systems but which is often neglected in the planning stages of development. Figure 1 shows the Operational Complex of the System divided into input, processing, and output phases, together with the data flow between those phases. The data base maintenance activity shown in Figure 1 is one of the activities accomplished in the Support Complex of the System. The Operational Complex of the System consists of 10 Technology Subsystems and 4 Executive Subsystems. The Support Complex of the System consists of four subsystems.

The elements of the System software below subsystem in the hierarchy are, from the top down: package, subpackage, and module. At the lowest level of the hierarchy is the basic building block of the System--the module. A module is the equivalent of a FORTRAN subprogram. Modules are subject to constraints of



**Figure 2. Top-Level Hierarchy of the System**

programming standards, providing management control and limiting maintenance costs. In the middle levels of the hierarchy, between subsystem and module, are the packages and subpackages. These have been identified as Computer Program Configuration Items (CPCIs) for the Predesign Phase.

For convenience, the collection of Technology Subsystems is called the Technology Component, and the collection of Executive Subsystems is called the Executive Component.

The order of execution of the software elements of the Technology Component and the Executive Component and the selection of data used during an analysis run are specified by the Sequence Control Table, which is constructed by the User Input Package (a package of the Executive Component) from user input for an analysis run. ("Software element" is a general phrase used to denote a member at any level of the System hierarchy, usually a package or a subpackage; because a package or a subpackage may in certain limited instances consist of only one module, a software element can denote a module.)

Communication between the software elements listed in the Sequence Control Table is accomplished by allowing a software element to use input data that were calculated by another software element earlier in the execution sequence. Technology Component software elements have the capability to affect the execution sequence dynamically by issuing to the Executive Component a System Command to be executed immediately. Thus, a software element is able to issue a command causing the Executive Component to execute a second software element or select additional data from the Run Data Base. Upon completion of the System Command execution, the software element issuing the command continues to operate from the point immediately following the point at which the command was issued.

The Technology Component is that part of the System that defines the mathematical analysis for the entire simulation. Although 10 subsystems are currently defined,



the design of the Technology Component is not rigid; i.e., if a new capability is defined in the Type A System Specification, either it is allocated to an existing subsystem or a new subsystem is defined to accommodate it. The principal reasons for defining subsystems are to provide a management control tool and to allow for unity and ease of documentation. Because of the ability of the Executive Component to recognize software elements independent of affiliation with a particular subsystem, subsystems may be added, deleted, or reorganized completely without affecting operation of the System.

The executive and supervisory control of System data and software during an analysis run is localized in the Executive Component of the Operational Complex. The Executive Component establishes the analyst's interface with the Operational Complex, monitors and controls the execution of the software elements during an analysis run, manages and controls the data needed to analyze a helicopter configuration, and provides a computer-independent interface to host operating system services. The centralization of executive and supervisory functions minimizes the complexity and extent of the interfaces among the software elements of the Technology Component. This permits software elements of the Technology Component to be developed independently and in parallel by multiple subcontractors. The separation of executive and supervisory functions from Technology Component functions ensures that the goals of maximum System transportability and minimum computer or operating system dependency are realized.

#### **1.3.4 A User's Overview of System Data**

System data are divided into three categories: data input to the System, data used by more than one software element within the System, and data output from the System.

Data input to the System consist of three principal types: the Master Data Base, the user input, and the Master Command File. The Master Data Base contains

helicopter-oriented data likely to be used frequently in analysis runs. The Master Data Base consists of sets of data that can be hierarchically arranged to represent physical configurations, such as aircraft or components of aircraft, to represent flight conditions or maneuvers, and to represent failure or damage to the physical configuration.

The user input data are the collection of statements designed for use by a person with no knowledge of the design of the System but with knowledge of the physical configuration to be analyzed. The form of the user input emphasizes the description of the configuration, flight conditions, failure/damage effects, and results to be output. Through the user input data, the Particular Functional Capability specified in the Type A System Specification is accessed.

The Master Command File contains sequences of System Commands that are the mechanism for controlling the steps of the analysis. The System Commands are of two types: the execution command, which causes a specific software element to be executed; and the sequence control command, which conditionally causes a transfer of control within the sequence of System Commands. The direct creation and modification of sequences of System Commands by the user, e.g., the methods developer, represents the General Functional Capability of the System.

Data used by more than one software element within the System are contained in the Run Data Base. The Run Data Base is initialized during the input phase of an analysis run to contain all input data required for the processing phase. It also contains all sets of data that are generated by one software element and used by another during the course of the analysis run. The Run Data Base, unlike the Master Data Base, is a temporary data base that varies from run to run and is not saved at the end of an analysis run.

Data output from the System consist primarily of printed reports and plots of the results of the analysis. Output may also include prints or plots of intermediate

results and information such as intermediate results stored in restart files for potential use in subsequent analysis runs.

#### **1.4 MAJOR SYSTEM DESIGN CONSIDERATIONS: USABILITY, EFFICIENCY, TRANSPORTABILITY, AND EXTENDABILITY**

For the System to be universally accepted by the helicopter analysis community, it must be user-oriented, efficient, transportable, and extendable. The CSC/BHT design synthesizes a System having all four of these necessary characteristics.

##### **1.4.1 Usability**

The most frequent use of the System is to perform a straightforward analysis of a physical configuration similar to one in the Master Data Base. The Master Data Base is a collection of data at each installation that describes aircraft, aircraft components, and other analysis components that may be analyzed; maneuvers, conditions, and operating regimes for an analysis; and failure/damage effects that might be considered. It is the intent of the System design that data which are used repeatedly in analyses at an installation will be maintained in the Master Data Base by personnel at the installation. Each helicopter firm and Government agency at which the System is installed can configure the Master Data Base to suit its own needs. Flight conditions, maneuvers, and operating regimes from the Master Data Base can be used with or without failure/damage effects. For this mode of using the System, a small user input data deck suffices to specify the desired analysis.

To provide the General Functional Capability required by the Baseline Type A System Specification, the engineering user or methods developer may create directly System Commands in any combination and in any order (a System Command identifies the software element to be executed and defines its required input and output). To produce meaningful results however, the input required by a software element must have been calculated prior to the time of software element



execution. Thus, the use of the System in this manner requires a knowledge of the data produced by, and required by, the software element.

The System may be used from an interactive terminal to prepare valid input data, to examine previously calculated results, and to execute any portion (or all) of the engineering analysis in an interactive mode.

Other uses of the System are to support the engineer's use of the System. The primary supporting use of the System is to maintain the Master Data Base. Each installation will make its own rules for Master Data Base maintenance, but the recommended procedure is to test all data before including it in the Master Data Base and to include in the Master Data Base any data that are likely to be used repeatedly.

There are many other ways to use the System in support of its primary use. For example, upon request the System will predict the cost of making an analysis run to assist the user in making efficient use of the System. For another example, the System can be used to make changes in System Command Sequences in the Master Command File.

As users become more familiar and more confident in the analysis capabilities provided by the System, enhancements to the user's interface with the System will inevitably be recommended. To facilitate incorporating these enhancements, the direct interface to the user is defined in one subsystem of the Executive Component, namely the User Interface Subsystem. This isolation of the direct user interface from the rest of the System software permits the user's interface to be modified without requiring attendant changes to other software in the System.

#### 1.4.2 Efficiency

The System is designed to efficiently analyze both small problems and large problems. The size of the problem is not restricted by the amount of computer memory available to the System. However, unlike NASTRAN, NASA's Structural

Analysis System, the Second Generation Comprehensive Helicopter Analysis System is also designed to efficiently analyze small problems. In NASTRAN, all problems are assumed to be large, thus penalizing the efficiency of analyzing small problems. In the Second Generation Comprehensive Helicopter Analysis System, two design features are included which emphasize the efficiency of small problem analyses. First, the Run Data Base may be entirely memory-resident (for small problems), may be entirely resident on peripheral storage device (for large problems), or may be partially memory-resident and partially peripheral device resident (for intermediate-size problems). Second, the data base management services provided by the Data Base Management Subsystem of the Executive Component are designed to eliminate any dependency of Technology Component software elements on the location (i.e., memory or peripheral device) of data within the Run Data Base. Thus, the fixed input/output overhead penalty incurred by small problems in NASTRAN is avoided in the Second Generation Comprehensive Helicopter Analysis System.

#### 1.4.3 Transportability

The System is designed to be transportable, that is, to be transferred to a different computer with a different operating system at relatively low cost and in a relatively short time compared to the time ordinarily required to convert a system from one computer to another. This will be accomplished in the Development Phase when both the First Level Release and the Second Level Release are transported from the Host 1 computer (IBM S/370 and S/360 series) to the Host 2 computer (CDC 6000 and CYBER series). Having both releases of the System available on the two computer systems most widely used by the helicopter analysis community will contribute to the System's acceptance by both helicopter firms and Government agencies.

The identification of the Operating System Service Subsystem as one of the Executive Subsystems is the aspect of the design that allows easy transportability of the

System. All operating system services for the remainder of the System are performed by the Operating System Service Subsystem. The interface between the Operating System Service Subsystem and the remainder of the System is computer independent and is thus unchanged when the System is moved to a new computer. When the System is moved to a new computer, the Operating System Service Subsystem changes to use the new host operating system without changing its interface with the rest of the System. Because that interface is not changed, the remainder of the System need not be changed.

#### 1.4.4 Extendability

The system is designed to be extendable so that a methods developer can easily experiment with new or improved analysis capabilities. The primary features of the System design that accomplish this goal are (1) the division of the System into software elements with clearly defined functions and interfaces, (2) the separation of the flow of control from the logic of individual software elements, and (3) the separation of data management from the logic of individual software elements.

The definitions of the function performed by a software element and its interface with the rest of the System allow easy substitution of an improved software element as helicopter analysis and/or software technology advances. If the new software element requires a different interface, the extent of change required in other software elements can be accurately assessed.

The flow of control for a helicopter analysis is defined by an internal sequence of commands in the Sequence Control Table rather than in the logic of the software elements. Substitution of improved (e.g., more accurate, more efficient) software elements or introduction of new software elements over the life of the System thus does not require the rewriting of other software elements to change the flow of control to include the new software element. Only a sequence of System Commands need be modified to include the new software element.



The data required by a software element are specified by name rather than by physical location on a device or by relative location within a record. Rearrangement of data as necessary or convenient during the life of the System is accomplished without change to the software elements provided the data names are retained. Reasons for data rearrangement during the life of the System include making efficient use of a particular data storage device and adding data required by or produced by a new software element implementing an improved analysis technique.

## SECTION 2 - SYSTEM DESIGN SUMMARY

This section presents a summary of the System designed by Computer Sciences Corporation (CSC) and Bell Helicopter Textron (BHT) to meet the requirements of the Baseline Type A System Specification for the Second Generation Comprehensive Helicopter Analysis System, CSC Document CSC/SD-78/6007. Section 2.1 presents the mathematical basis for the System. Section 2.2 defines the terms used to describe the different levels of the software system hierarchy. The Operational Complex, which is the part of the System that contains the software that solves helicopter analysis problems, is discussed in Section 2.3. The Support Complex, which aids the development, test, configuration management, and documentation of the System, is discussed in Section 2.4. The section concludes with a discussion in Section 2.5 of the System Command Sequences for the five aircraft technical characteristics of performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability.

### 2.1 MATHEMATICAL BASIS FOR THE SYSTEM

Three goals were established in selecting the mathematical basis for the System design: (1) provide a single unifying basis for the analytical solution of helicopter analysis problems; (2) provide a single unifying concept for representing both helicopter configuration components and aerodynamic effects; and (3) provide a method for analyzing a physical configuration composed of independently defined components. A unified approach has been formulated which provides a consistent representation of the differential equations required for helicopter analysis and simulation, a consistent method for representing physical components (namely, finite elements), and a systematic method for coupling components which represent a physical configuration. Section 2.1.1 contains a discussion of the type of problems that are to be solved by the System and the mathematical solution of these problems. Section 2.1.2 discusses the applicability of the finite element concept to the representation of helicopter configurations. Section 2.1.3 deals with the

important problem of coupling of components; to reduce computer costs, the method of component modes was incorporated in the design for the First Level Release of the System. Section 2.1.4 discusses the equally important problem of aerodynamic effects. Section 2.1.5 presents numerical analysis considerations important to the reliability and acceptability of the System. Finally, Section 2.1.6 describes how the modularity and adaptability of the System design accommodate innovative analysis concepts related to helicopter technology without necessitating major design modifications and at relatively little cost.

#### 2.1.1 Type of Problems To Be Solved by the System

The System to be developed is required to predict, accurately and reliably, the performance, stability and control, loads and vibrations, acoustics, and aero-elastic stability characteristics of rotary-wing aircraft configurations. The proposed System must therefore be capable of solving all problems associated with these five aircraft technical characteristics.

In its most general form, helicopter analysis and simulation can be represented by a system of second-order differential equations in the form of the following matrix equation:

$$[M] \{\ddot{q}\} + [C] \{\dot{q}\} + [K] \{q\} = \{F\} \quad (2)$$

where  $\{q\}$ ,  $\{\dot{q}\}$ , and  $\{\ddot{q}\}$  are generalized displacement, velocity, and acceleration vectors, respectively;  $[M]$ ,  $[C]$ , and  $[K]$  are the mass, damping, and stiffness matrices, respectively, of the aircraft configuration under analysis (these matrices may be constant, periodic, or otherwise time variant, or weak functions of  $\{q\}$  and  $\{\dot{q}\}$ ); and  $\{F\}$  is a vector of generalized forces that may be constant but is more commonly periodic or otherwise time variant. In addition,  $\{F\}$  may be a linear or nonlinear function of the  $\{q\}$  and  $\{\dot{q}\}$  vectors.



For numerical processing, Equation (2) can be transformed into a set of first-order differential equations using the following transformation:

$$\begin{aligned} \{p_1\} &= \{q\} \\ \{p_2\} &= \{\dot{q}\} = \{\dot{p}_1\} \end{aligned} \quad (3)$$

Using these transformations in Equation (2) yields

$$\begin{bmatrix} I & 0 \\ 0 & M \end{bmatrix} \begin{Bmatrix} \dot{p}_1 \\ \dot{p}_2 \end{Bmatrix} = \begin{bmatrix} 0 & I \\ -K & -C \end{bmatrix} \begin{Bmatrix} p_1 \\ p_2 \end{Bmatrix} + \begin{Bmatrix} 0 \\ F \end{Bmatrix} \quad (4)$$

where  $I$  is the identity matrix.

In general, transformations of the type in Equation (3) can be employed to transform any given set of higher order differential equations into a set of first-order differential equations of the type in Equation (4). In this way, the System can be adapted for the numerical processing of higher order differential equations also.

The degrees of freedom represented by the  $\{q\}$  vector mentioned above are called the analysis degrees of freedom. These can represent a variety of physical or convenient mathematical quantities. In the analysis of structural components, these can represent either physical displacements at the node points or modal coordinates. Other types of analysis degrees of freedom can include fluid pressures and flow rates for a hydraulic actuator or control motions and derivatives for an electronic flight control system.

All helicopter problems to be analyzed by the System can be represented either by Equation (2) or by derivations thereof. Problems that require time-history solutions, such as aircraft maneuvers with prescribed control motions or prescribed

responses, and flight simulations involving atmospheric disturbances or failure/damage effects, are examples of problems that can be represented by the general form of Equation (2). A time-history solution is obtained by a direct numerical integration of Equation (2).

Basic to many helicopter analysis problems is the determination of the steady-state flight or trim for various prescribed flight conditions. Many of the conditions of interest for performance, stability and control, loads and vibrations, and acoustics are steady-state trim conditions. Several critical helicopter design considerations which can be analyzed during a steady-state flight condition include the effects of airspeed variation on power required, control positions, oscillatory bending moments or stresses, cabin area vibrations, and perceived noise levels inside the aircraft as well as on the ground.

In most cases, steady-state flight can be regarded as the response to a force input that varies harmonically. The matrix equation for steady-state flight can therefore be represented by

$$[M]\{\ddot{q}\} + [C]\{\dot{q}\} + [K]\{q\} = \{f\} \sin \omega t \quad (5)$$

which is obtained from Equation (2) by setting  $\{F\} = \{f\} \sin \omega t$ , where the vector  $\{f\}$  is a function of  $\{q\}$  and  $\{\dot{q}\}$  as well as the flight conditions,  $\omega$  is the forcing frequency, and  $t$  represents time.

Because  $\{f\}$  is a function of  $\{q\}$  and  $\{\dot{q}\}$ , the trim condition problem represented by Equation (5) requires iterative techniques in order to be able to converge to the trimmed state. Before a trim computation can be attempted, certain assumptions about the form of the solution must be made based on the complexity of

the matrix equations selected for the solution. In the general case, the trim solution can be assumed to be a truncated Fourier series solution in which the fundamental frequencies are the rotational speeds of the rotors and the number of harmonics to be computed is specified by the user. This is represented by

$$\begin{aligned} \{q\} = \{q_0\} &+ \left\{q_{c_1}\right\} \cos \omega t + \left\{q_{s_1}\right\} \sin \omega t \\ &+ \left\{q_{c_2}\right\} \cos 2 \omega t + \left\{q_{s_2}\right\} \sin 2 \omega t + \dots \\ &+ \left\{q_{c_n}\right\} \cos n \omega t + \left\{q_{s_n}\right\} \sin n \omega t \end{aligned} \quad (6)$$

where  $n$  is the number of harmonics specified by the user. The coefficients of the harmonics are determined by substituting Equation (6) into Equation (5) and using the method of undetermined coefficients.<sup>1</sup> For a simple case, only the response due to the constant term and the first harmonic need be considered. This reduces the number of periodic coefficients in the matrix equations, thus simplifying the solution. For the most elementary case, the periodic coefficients can be neglected entirely so that only the response due to the constant term is calculated.

In the case of highly nonlinear mathematical models, the assumption of harmonic response may be significantly in error. For these circumstances, the design includes a fly-to-trim option. This option is designed to calculate the true nonlinear response for a steady-state flight condition in a manner very similar to a flight test procedure.

---

<sup>1</sup>Kreyszig, E., ADVANCED ENGINEERING MATHEMATICS, New York, John Wiley and Sons, 1967.



Another fundamental problem in helicopter analysis is the determination of the natural frequencies and mode shapes of the aircraft structure and its components. This problem is represented by the simple and familiar equation

$$[M]\{\ddot{q}\} + [K]\{q\} = \{0\} \quad (7)$$

which is obtained from Equation (2) by setting  $[C] = [0]$  and  $\{F\} = \{0\}$ . The solution to this equation yields eigenvalues and eigenvectors (natural frequencies and mode shapes) of the component or configuration in a vacuum (without air loads).

Stability and control analysis and aeroelastic stability analysis can both be performed by considering small perturbed motions about a trimmed flight condition. The primary difference between stability and control analysis and aeroelastic stability analysis lies in the level of detail and the type of configuration considered. Stability and control analysts are primarily interested in rigid-body aircraft motions and responses to cockpit control motions. The handling qualities of the aircraft are judged by the rigid-body behavior of the fuselage whether the simulation model used is a simple one or a fully aeroelastic representation.

On the other hand, in the case of aeroelastic stability analysis, complex dynamic coupling and unsteady aerodynamics are the essential ingredients, and the airframe itself may be of secondary importance. Investigations of classical flutter and stall flutter on a rotor are examples of aeroelastic stability analyses that are independent of the airframe. In contrast, ground resonance and whirl flutter are aeroelastic stability problems that involve both rotors and airframe. As a final example, the phenomenon of air resonance is one in which the fields of aeroelastic stability and stability and control truly intertwine.

Stability analysis involves the computation of stability derivatives,<sup>2,3</sup> some of which are calculated by using Equation (2). The derivatives may be calculated either analytically or numerically. These stability derivatives are then used to set up locally linearized equations of motion of the form

$$[M']\{\ddot{q}\} + [C']\{\dot{q}\} + [K']\{q\} = \{F'\} \quad (8)$$

where coefficient matrices  $[M']$ ,  $[C']$ , and  $[K']$  and vector  $\{F'\}$  involve stability derivatives. The coefficient matrices in this equation are constant for classical stability analysis and periodic for Floquet analysis.

Stability analysis normally leads to the standard eigenvalue problem represented by the equation

$$[A - \lambda I] \begin{Bmatrix} p_1 \\ p_2 \end{Bmatrix} = \{0\} \quad (9)$$

where  $[A]$  is a square matrix,  $\lambda$  is a scalar quantity (real or complex), and  $[I]$  is a unit matrix. The eigenvalues (the  $\lambda$ s) of  $[A]$  characterize the stability of the physical configuration. Mode shapes corresponding to these characteristic values indicate the type of motion associated with each stable or unstable eigenvalue.

Aircraft maneuvers with prescribed controls or prescribed responses and flight simulations during which there are sudden changes of atmospheric conditions or sudden structural changes caused by loss of components or failure/damage effects

<sup>2</sup>Seckel, E., STABILITY AND CONTROL OF AIRPLANES AND HELICOPTERS, New York, Academic Press, 1968.

<sup>3</sup>Bramwell, A. R. S., HELICOPTER DYNAMICS, New York, John Wiley and Sons, 1976.

are examples of problems that can be represented by the general form given by Equation (2). The solutions to these transient problems require the generation of time histories of the vector  $\{q\}$  by numerical integration techniques. To start these time-history solutions, it is first necessary to have a realistic set of initial conditions. These initial conditions may be obtained either from user input or from the results of a previously calculated trim condition.

Predictions of performance, loads and vibrations, and acoustics characteristics need not be regarded as special mathematical procedures because these characteristics can be determined as a byproduct of the normal processing for a trimmed flight condition or at time points during a transient solution.

#### 2.1.2 Finite Element Concept

The generation of the mass, damping, and stiffness matrices shown in Equation (2) can be a very complex process for a complex configuration like a helicopter. The finite element concept, which is a very systematic and unifying method of analysis, lends itself very well to reducing this complexity. This method of analysis<sup>4,5,6</sup> has been successfully employed in structural analysis applications for many years.

---

<sup>4</sup>Desai, C. S., and Abel, J. F., INTRODUCTION TO THE FINITE ELEMENT METHOD, New York, Van Nostrand Reinhold Company, 1972.

<sup>5</sup>Brebbia, C. A., and Connor, J. J., FUNDAMENTALS OF FINITE ELEMENT TECHNIQUES, New York, John Wiley and Sons, 1974.

<sup>6</sup>Gallagher, R. H., FINITE ELEMENT ANALYSIS FUNDAMENTALS, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1975.



In recent years, the suitability and adaptability of this method for the analysis of both nonrotating and rotating components of a helicopter have been demonstrated.<sup>7,8,9</sup>

The use of the finite element concept is the basis of the System design. This unifying and systematic concept makes it possible for the System to analyze varieties of helicopter configurations and designs without radically changing the basic structure of the System. This concept also makes it possible to analyze complex individual helicopter structures (such as a complete fuselage, a complete rotor, and an engine or an electronic flight control system) independently before they are combined to form the aircraft configuration. This concept also permits the use of specialized analysis techniques for handling particular components of a helicopter.

The design treats the aircraft configuration as a collection of aircraft and other analyses components that are represented by finite elements connected together at specified locations called node points. The behavior of the total configuration is then derived from the analysis of the individual components that constitute the configuration.

The degrees of freedom involved in the analysis (the components of the  $\{q\}$  vector in Equation (2)) are called the analysis degrees of freedom. In the direct method of finite element analysis, these degrees of freedom are the physical

---

<sup>7</sup> Cronkhite, J. D., DEVELOPMENT, DOCUMENTATION AND CORRELATION OF A NASTRAN VIBRATION MODEL OF THE AH-1G HELICOPTER AIRFRAME, NASTRAN: User's Experiences, NASA TM X-3428, October 1976, pp. 273-294 (see also Reference 8).

<sup>8</sup> Pamidi, P. R., and J. D. Cronkhite, ADDITION OF RIGID ELEMENTS TO NASTRAN, Sixth NASTRAN Users' Colloquium, NASA Conference Publication 2018, October 1977, pp. 449-468.

<sup>9</sup> Krishna Murthy, A. V., and Sridhara Murthy, S., FINITE ELEMENT ANALYSIS OF ROTORS, Mechanism and Machine Theory, Vol. 12, 1977, pp. 311-322.

displacements at the node points. In the modal method of finite element analysis, these degrees of freedom are the modal coordinates.

The node points employed may be either dynamic, aerodynamic, or both. Providing this capability can reduce computing costs by avoiding unnecessary aerodynamic calculations at points where the airloads are insignificant and unnecessary dynamic calculations at points where dynamic effects are insignificant. (In aerodynamic applications, the best aerodynamic model is normally defined quite differently from the structural dynamic model with which it is connected. Section 2.1.4 discusses how aerodynamic effects can be represented by using aerodynamic models.)

### 2.1.3 Coupling of Components

The System must be capable of handling a variety of aircraft and other components combined to form a variety of configurations. A general and systematic method for coupling various components is therefore crucial to the success and acceptability of the System.

Coupling of components (sometimes called dynamic coupling) can be regarded as a logical extension of the finite element concept discussed in the previous section. Thus, just as finite elements are combined to result in a structure or component, structures or components in turn can be dynamically coupled to yield complex structures or configurations.

A general and systematic method for coupling components reduces considerably the amount of work required to combine components that have already been separately derived and tested. This procedure of combining components is particularly applicable to the coupling of rotating components with components of the fixed system and is therefore well suited to the treatment of the helicopter analysis problem.

To effectively serve the needs of the helicopter analysis community, the method of coupling employed in the System must satisfy the following three important criteria:

1. It must employ a minimum number of degrees of freedom.
2. It must permit totally independent analysis of the components.
3. It must be compatible with test procedures in order to facilitate comparison with test results.

Consider a configuration made up of  $n$  components. The behavior of any component  $i$  can then be represented by the matrix differential equation

$$[M_i] \{\ddot{p}_i\} + [C_i] \{\dot{p}_i\} + [K_i] \{p_i\} = \{F_i\} \quad (10)$$

where  $[M_i]$ ,  $[C_i]$ , and  $[K_i]$  are the mass, damping, and stiffness matrices of component  $i$ ;  $\{F_i\}$  represents the forces acting on the component; and  $\{p_i\}$  represents the degrees of freedom associated with the component.

The following matrix equation represents the mass, damping, and stiffness matrices for all the components in an uncoupled configuration:

$$\begin{bmatrix} [M_1] & & & \\ & [M_2] & & \\ & & \ddots & \\ & & & [M_i] & \\ & & & & \ddots & \\ & & & & & [M_n] \end{bmatrix} \begin{Bmatrix} \{p_1\} \\ \{p_2\} \\ \vdots \\ \{p_i\} \\ \vdots \\ \{p_n\} \end{Bmatrix} + \begin{bmatrix} [C_1] & & & \\ & [C_2] & & \\ & & \ddots & \\ & & & [C_i] & \\ & & & & \ddots & \\ & & & & & [C_n] \end{bmatrix} \begin{Bmatrix} \{\dot{p}_1\} \\ \{\dot{p}_2\} \\ \vdots \\ \{\dot{p}_i\} \\ \vdots \\ \{\dot{p}_n\} \end{Bmatrix} + \begin{bmatrix} [K_1] & & & \\ & [K_2] & & \\ & & \ddots & \\ & & & [K_i] & \\ & & & & \ddots & \\ & & & & & [K_n] \end{bmatrix} \begin{Bmatrix} \{p_1\} \\ \{p_2\} \\ \vdots \\ \{p_i\} \\ \vdots \\ \{p_n\} \end{Bmatrix} = \begin{Bmatrix} \{F_1\} \\ \{F_2\} \\ \vdots \\ \{F_i\} \\ \vdots \\ \{F_n\} \end{Bmatrix} \quad (11)$$



which, in condensed form, can be represented as

$$[M_u] \{\ddot{p}\} + [C_u] \{\dot{p}\} + [K_u] \{p\} = \{F_u\} \quad (12)$$

Equations (11) and (12) represent a disjoint or uncoupled configuration made up of the  $n$  components. Coupling the components implies certain relationships among the vectors  $\{p_1\}$ ,  $\{p_2\}$ , ...,  $\{p_i\}$ , ..., and  $\{p_n\}$ . These relationships can be expressed by a matrix relationship of the form

$$\begin{Bmatrix} \{p_1\} \\ \{p_2\} \\ \vdots \\ \{p_i\} \\ \vdots \\ \{p_n\} \end{Bmatrix} = \{p\} = \begin{Bmatrix} [\beta_1] \\ [\beta_2] \\ \vdots \\ [\beta_i] \\ \vdots \\ [\beta_n] \end{Bmatrix} \{q\} = [\beta] \{q\} \quad (13)$$

where  $\{q\}$  is the vector that represents the degrees of freedom of the coupled configuration as in Equation (2), and any partition  $[\beta_i]$  of  $[\beta]$  is a transformation matrix that relates the degrees of freedom  $\{p_i\}$  of component  $i$  to the degrees of freedom  $\{q\}$  of the coupled configuration. Normally, the degrees of freedom represented by  $\{q\}$  for the coupled configuration represent a subset of the degrees of freedom represented by  $\{p\}$  for the uncoupled configuration.  $[\beta]$  can be assumed to be time invariant without any loss of generality.

This is not an undue restriction on the modeling because transformations associated with rotating coordinate systems are performed within individual component representations before such components are coupled together.

Substitution of Equation (13) in Equation (12) and premultiplication of the resultant relationship by  $[\beta]^T$  yield the matrix equation

$$[M] \{\ddot{q}\} + [C] \{\dot{q}\} + [K] \{q\} = \{F\} \quad (14)$$

where the coefficient matrices are given by

$$[M] = [\beta]^T [M_u] [\beta]$$

$$[C] = [\beta]^T [C_u] [\beta]$$

$$[K] = [\beta]^T [K_u] [\beta]$$

and

$$\{F\} = [\beta]^T \{F_u\}$$

Note that Equation (14) is the same as Equation (2).

Coupling of components can be accomplished by two different methods. The first method is commonly referred to as substructure analysis. One of the earliest comprehensive discussions of this method was presented by Przemieniecki.<sup>10</sup>

The second method, commonly called the method of component modes, was first

<sup>10</sup> Przemieniecki, J. S., MATRICES STRUCTURAL ANALYSIS OF SUBSTRUCTURES, AIAA Journal, Vol. 1, No. 1, January 1963, pp. 138-147.

proposed by Hurty.<sup>11,12</sup> This method and others derived from it have since received much attention in the aerospace industry.<sup>13 through 25</sup>

- <sup>11</sup> Hurty, W. C., VIBRATIONS OF STRUCTURAL SYSTEMS BY COMPONENT MODE SYNTHESIS, Proc. ASCE, Journal of the Engineering Mechanics Division, August 1960, pp. 51-69.
- <sup>12</sup> Hurty, W. C., DYNAMIC ANALYSIS OF STRUCTURAL SYSTEMS USING COMPONENT MODES, AIAA Journal, Vol. 3, No. 4, April 1965, pp. 678-685.
- <sup>13</sup> Gladwell, G. M. L., BRANCH MODE ANALYSIS OF VIBRATING SYSTEMS, Journal of Sound and Vibration, Vol. 1, 1964, pp. 41-59.
- <sup>14</sup> Bamford, R. M., A MODAL COMBINATION PROGRAM FOR DYNAMIC ANALYSIS OF STRUCTURES, NASA TM 33-290, Jet Propulsion Laboratory, 1966.
- <sup>15</sup> Bajan, R. L., and Feng, C. C., FREE VIBRATION ANALYSIS BY THE MODAL SUBSTITUTION METHOD, American Astronautics Society Symposium, Paper No. 68-8-1, July 1968.
- <sup>16</sup> Benfield, W. A., and Hruda, F. R., VIBRATION ANALYSIS OF STRUCTURES BY COMPONENT SUBSTITUTION, AIAA Journal, Vol. 9, July 1971, pp. 1255-1261.
- <sup>17</sup> Hintz, R. M., ANALYTICAL METHODS IN COMPONENT MODAL SYNTHESIS, AIAA Journal, Vol. 13, August 1975, pp. 1007-1016.
- <sup>18</sup> Craig, R. R., Jr., and Bampton, M. C. C., COUPLING OF SUBSTRUCTURES FOR DYNAMIC ANALYSES, AIAA Journal, Vol. 6, No. 7, July 1968, pp. 1313-1319.
- <sup>19</sup> MacNeal, R. H., A HYBRID METHOD OF COMPONENT MODE SYNTHESIS, Computers and Structures, Vol. 1, December 1971, pp. 581-601.
- <sup>20</sup> Rubin, S., AN IMPROVED COMPONENT-MODE REPRESENTATION, AIAA Paper No. 74-386, presented at AIAA/ASME/SAE 15th Structures, Structural Dynamics and Materials Conference, Las Vegas, Nevada, April 17-19, 1974.
- <sup>21</sup> Gleseke, R. K., ANALYSIS OF NONLINEAR STRUCTURES VIA MODE SYNTHESIS, NASTRAN: Users' Experiences; NASA TM X-3278, September 1975, pp. 341-360.
- <sup>22</sup> Klosterman, A. L., A COMBINED EXPERIMENTAL AND ANALYTICAL PROCEDURE FOR IMPROVING AUTOMOTIVE SYSTEM DYNAMICS, Society of Automotive Engineers, Paper No. 720093, January 1972.
- <sup>23</sup> McClelland, W. A., and Klosterman, A. L., USING NASTRAN FOR DYNAMIC ANALYSIS OF VEHICLE SYSTEMS, Society of Automotive Engineers, Paper No. 740326, March 1974.
- <sup>24</sup> Herting, D. N., and Hoesly, R. L., DEVELOPMENT OF AN AUTOMATED MULTI-STAGE MODAL SYNTHESIS SYSTEM FOR NASTRAN, Sixth NASTRAN Users' Colloquium, NASA Conference Publication 2018, October 1977, pp. 435-448.
- <sup>25</sup> Wilson, T. L., A NASTRAN DMAP ALTER FOR THE COUPLING OF MODAL AND PHYSICAL COORDINATE SUBSTRUCTURES, Sixth NASTRAN Users' Colloquium, NASA Conference Publication 2018, October 1977, pp. 119-130.



In both of these methods, a complex structure is considered to be made up of component substructures connected at specified node points, and the behavior of the structure is determined by considering the behavior of the component substructures at the connection node points. The essential difference between the two methods lies in the different manner in which the behavior of the component substructures (namely, the vectors  $\{p_1\}$ ,  $\{p_2\}$ , ... in Equations (11) and (13)) is represented. In substructure analysis, the component substructures are represented by the physical displacements at the connecting node points, whereas, in the component modes method, the substructures are represented by their modal data (eigenvalues and eigenvectors). The chief advantage of the component modes method over the substructure analysis method is that reliable results can normally be obtained using significantly fewer degrees of freedom than are needed in substructure analysis. This advantage leads to cost savings. Also, the component modes method does not require that the modal data be generated only by analytical methods; some or all data can be derived from experimental data.<sup>22</sup>

Several approaches to the use of component modes have been proposed by various investigators. The approach suggested by Hurty,<sup>11, 12</sup> and variations of it proposed by others,<sup>14 through 17</sup> employ rigid-body modes, constraint modes, and normal modes with fixed constraints. These approaches are incompatible with helicopter test procedures and therefore make it quite difficult to compare analysis results with test data. The approaches proposed by MacNeal<sup>19</sup> and Rubin<sup>20</sup> employ free-body modes and residual effects. Both MacNeal's and Rubin's approaches give good accuracy (this is particularly so for Rubin's approach, which is an improvement over MacNeal's approach), but both are restrictive and cumbersome in formulation. More recently, a general component modes approach for inclusion in NASTRAN has been developed under NASA sponsorship.<sup>24</sup> This approach has the accuracy of Rubin's method<sup>20</sup> without the restriction of having to use specific types of modes. In addition, all other methods mentioned above can be regarded as special cases of this general approach.

This component modes approach also satisfies all the three criteria mentioned earlier for a systematic method of coupling components. Therefore, the component modes approach developed for inclusion in NASTRAN is the approach best suited for inclusion in the System.

The System design can accommodate either of the two dynamic coupling methods mentioned above (i.e., substructure analysis or component modes). However, the funds available for the Development Phase may not permit the implementation of both methods. Because of the advantages offered by the component modes method, CSC recommends the implementation of this method of coupling for the First Level Release of the System. If funding permits, it is recommended that the substructure analysis method be added to the Second Level Release of the System. It is also recommended that the simultaneous use of both these methods<sup>25</sup> be considered for inclusion in the Long Range System.

#### 2.1.4 Aerodynamic Effects

The proper treatment of the aerodynamic flow field and its effect upon the various components of the helicopter is one of the most important requirements of the System. As a starting point, it must be realized that the air mass surrounding the aircraft is a continuum that comes into physical contact with every node point of the aircraft. Thus, for the most rigorous analysis (e.g., for research applications), the coupling of every node point with every other node point through the aerodynamic velocity and pressure field must be considered. In addition, the inherent nonlinearities of both steady and unsteady aerodynamics in subsonic and transonic flow must be considered.

In general, the aerodynamic effects cannot be easily accounted for by considering additional independent degrees of freedom in the  $\{q\}$  vector of Equation (2). The normal practice is to account for the aerodynamic effects by including them

in the  $\{F\}$  vector in Equation (2) as well as by relating them to the structural dynamic degrees of freedom in the  $\{q\}$  vector. However, for certain simplified models of induced velocity, it is possible to include, in the independent degrees of freedom, variables which describe the time-averaged induced velocity field and which approximate the lower harmonics of induced velocity. Also, in potential flow solutions, the element strengths and vortex positions are treated as analysis degrees of freedom.

Different applications require different simplifying assumptions about the aerodynamic effects. In particular, there is a sharp contrast between the aerodynamic analysis used for preliminary design and that used for research applications. To facilitate the application of differing assumptions about aerodynamic effects, two distinct models of aerodynamic effects have been identified: aerodynamic force models and aerodynamic interference models.

Aerodynamic force models are used to represent the forces exerted by the surrounding fluid on the aircraft structure. These models can represent non-linear analytic or tabular functions. A typical aerodynamic force model represents the steady lift, drag, and pitching moment for an aerodynamic node point on the rotor. The functional form for such a representation is

$$\begin{aligned} L_i &= L(\{q\}, \{\dot{q}\}, t) \\ D_i &= D(\{q\}, \{\dot{q}\}, t) \\ M_i &= M(\{q\}, \{\dot{q}\}, t) \end{aligned} \tag{15}$$

where  $i$  is an aerodynamic node point,  $L_i$  is the lift force,  $D_i$  is the drag force,  $M_i$  is the pitching moment, and the functions  $L$ ,  $D$ , and  $M$  include calculation of velocity components, aerodynamic angles, Mach number, dynamic pressure, and analytic or tabular evaluation of aerodynamic coefficients. It



should be emphasized that the aerodynamic force models can represent not only linear aerodynamics effects, but also nonlinear effects.

Aerodynamic interference models represent the direct aerodynamic coupling between aircraft components that occurs through the airmass. Aerodynamic interference models may represent one-way effects, as in the rotor-induced downwash on a wing. They may also represent mutual interference effects, as in the downwash effects for a tandem helicopter, which account for the effect of each rotor on the other simultaneously.

The functional form for an aerodynamic interference model that represents rotor downwash on a wing is as follows:

$$\bar{V}_{R/W} = \bar{V}(L_i, D_i, M_i \text{ \{for all } i \}, t) \quad (16)$$

where  $\bar{V}_{R/W}$  is the velocity vector for the flow induced by the rotor at the wing, and  $\bar{V}$  represents a function of the rotor forces and moments depending on the spatial relationship between the rotor and wing and allowing for a time delay for velocity changes.

Wake analysis and aerodynamic panel representations are other types of aerodynamic interference models.

#### 2.1.5 Numerical Analysis Considerations

While the numerical stability of integration procedures is very important to the reliability of the System, the Project team believes that other important areas of numerical analysis must be successfully addressed if the System is to be acceptable. Other important numerical analysis considerations include coordinate systems and transformations, stability and control and aeroelastic stability computations, solution of trim equations, interpolation, and statistical analysis and digital filtering.

The Project team has considered possible numerical analysis problems in two ways. First, the design of the System has been made independent of single numerical methods. Second, the combined experience of both companies has been used to address specific numerical methods for this application.

#### 2.1.5.1 Numerical Integration

The Project team has conducted much analysis in determining the "best" numerical integration techniques for application to flight dynamics problems. BHT has found that no predictor-corrector technique (such as Hamming's method, which was at one time implemented in C81) could tolerate the discontinuities in the forcing function encountered from a representation of a helicopter rotor in forward flight. The Runge-Kutta self-starting, averaging methods appear more appropriate for this application. Enhanced Runge-Kutta techniques that require fewer function evaluations and allow a variable integration interval (Runge-Kutta-Fehlberg) with no loss in accuracy or stability are available. Such enhancements can provide a significant increase in computational efficiency.

The direct numerical integration techniques employed in practice involve either explicit or implicit formulations; explicit formulations are based on equilibrium conditions at the previous time step, whereas implicit formulations are based on equilibrium conditions at the current time step. The central difference method is an example of an explicit method while the Newmark-Beta, Wilson-Theta, and Houbolt methods are examples of implicit methods.<sup>26</sup> The differences between explicit and implicit methods are quite important. While explicit methods typically require minimal storage and are computationally efficient, they are

---

<sup>26</sup> Bathe, K. J., and Wilson, E. L., NUMERICAL METHODS IN FINITE ELEMENT ANALYSIS, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1976.

hampered by the fact that their stability criteria limit the step sizes that can be employed. In general, these step-size limitations are related to the highest natural frequency of the system of equations under consideration. The implicit methods, on the other hand, offer unconditional stability at the expense of increased storage and reduced computational efficiency. However, implicit methods are particularly suitable for large systems of equations where the period of the highest natural frequency is generally not known.

Recently, an unconditionally stable explicit algorithm has been proposed for certain structural dynamics problems.<sup>27</sup> Other recent publications address the relative stability of various numerical integration techniques for vibration problems<sup>28</sup> and for transient rotor dynamics problems.<sup>29</sup>

#### 2.1.5.2 Coordinate Systems and Transformations

The analysis and simulation of a complex configuration like a helicopter necessarily involve the use of a large number of coordinate systems because each of the aircraft components and other analysis components is likely to have at least one coordinate system associated with it. Also, in many cases, coordinate systems can be associated with individual node points so that the behavior

---

<sup>27</sup> Trujillo, D. M., AN UNCONDITIONALLY STABLE EXPLICIT ALGORITHM FOR STRUCTURAL DYNAMICS, International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1579-1592.

<sup>28</sup> Wood, W. L., ON THE ZIENKIEWICZ FOUR-TIME-LEVEL SCHEME FOR THE NUMERICAL INTEGRATION OF VIBRATION PROBLEMS, International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1519-1528.

<sup>29</sup> Kascak, A. F., STABILITY OF NUMERICAL INTEGRATION TECHNIQUES FOR TRANSIENT ROTOR DYNAMICS, NASA Technical Paper 1092, National Aeronautics and Space Administration, November 1977.



at these points can be interpreted in a more meaningful manner. Further, many of the coordinate systems will be associated with rotating components such as rotor blades. The potential use of such a large number of coordinate systems therefore requires that a systematic, convenient, and general method be used for computing the related transformation matrices and for maintaining proper relationships among the various coordinate systems.

Many existing helicopter analysis programs employ Euler angles for locating coordinate systems and for defining the transformations among them. This method is easy to use and comprehend, but has the disadvantage that the transformations fail at certain singularity points. Such singularities have been encountered, for instance, in the simulations of helicopter looping maneuvers.

The sensitivity to singularities inherent in the use of Euler angles can be eliminated by the use of quaternion algebra,<sup>30,31</sup> which offers a systematic, general, and powerful method for handling different coordinate systems. It has also been demonstrated to be quite cost-effective for this purpose.

For this reason, all internal coordinate systems and coordinate transformations will be based on the use of quaternions; however, input and output will be in terms of Euler angles with which helicopter engineers are most familiar. The transformation between Euler angles and quaternions is given in Appendix E of Wertz.<sup>32</sup>

---

<sup>30</sup> Hamilton, W. R., ON A NEW SPECIES OF IMAGINARY QUANTITIES CONNECTED WITH A THEORY OF QUATERNIONS, Dublin Proc., Vol. 2, No. 13, November 1843, pp. 424-434.

<sup>31</sup> Yang, A. T., APPLICATION OF QUATERNION ALGEBRA AND DUAL NUMBERS TO THE ANALYSIS OF SPATIAL MECHANISMS, doctoral dissertation, Columbia University, New York, 1963.

<sup>32</sup> Wertz, J. R. (editor), SPACECRAFT ATTITUDE DETERMINATION AND CONTROL, Dordrecht-Holland, D. Reidel, in preparation.

#### 2.1.5.3 Stability and Control and Aeroelastic Stability Computations

Both the stability and control computations and the aeroelastic stability computations may require the solution of complex eigenvalue problems. In addition, the computation of natural frequencies requires an eigenvalue solution. Due to the advanced state of technology in this area and the development of the orthogonal reduction (e.g., Givens, Householder) and iteration methods (e.g., the QR algorithm), the Project team does not anticipate any serious problems in this area. BHT also has experience with Moving Block Fast Fourier Transform and Prony's Method for obtaining aeroelastic stability data from time-history records. These methods are not highly reliable at this time and must be used with some care.

Two new numerical methods for dealing with the stability of linear systems of periodic equations have recently been presented.<sup>33</sup> Based on the use of multi-variable Floquet-Liapunov theory, these methods have both been shown to be the most efficient, general, and practical numerical methods available at present. Another recent numerical contribution in the area of stability<sup>34</sup> should be studied further to determine its potential value to the System.

#### 2.1.5.4 Solution of Trim Equations

A demanding numerical problem occurs when solving the system of nonlinear algebraic equations that represent the helicopter trim condition. The complexity of the problem dictates the need for a successive approximation scheme such as the Newton-Raphson technique. This method, however, requires initial starting conditions and can have convergence difficulties. BHT experience with

---

<sup>33</sup> Friedmann, P., Hammond, C. E., and Woo, T. H., EFFICIENT NUMERICAL TREATMENT OF PERIODIC SYSTEMS WITH APPLICATION TO STABILITY PROBLEMS, International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1117-1136.

<sup>34</sup> Done, G. T. S., and Simpson, A., DYNAMIC INSTABILITY OF CERTAIN CONSERVATIVE AND NON-CONSERVATIVE SYSTEMS, Journal of Mechanical Engineering Science, Vol. 19, No. 6, 1977, pp. 251-263.



the multidimensional Newton-Raphson method in C81 has shown that the reliability of the technique can be significantly increased when it is augmented with limiters and numerical dampers. BHT believes that an enhanced Newton-Raphson technique can be made to work efficiently and reliably on the expanded systems of trim equations anticipated in the future for the System.

#### 2.1.5.5 Interpolation of Input Data

Performance analysis of aeronautical systems requires specification of numerous functions that characterize the aerodynamic coefficients and derivatives, inertial characteristics, propulsion parameters, and other subsystem and environmental effects. Such functions are frequently dependent on several independent variables and are stored as discrete points in tables. Interpolation routines must be provided in the software to extract the functional information from the tables. The interpolation routines must be fast because the tables are interpolated many times, must be reliable because key computations are dependent upon interpolated values, and must be flexible because the function being interpolated may differ significantly from problem to problem. Numerical analysts of CSC and BHT are acutely aware of the practical considerations of tabular interpolation. CSC has experience in developing reliable interpolators for aerodynamic data and suggests odd- and low- order polynomial interpolators (e.g., first and third order) capable of interpolating variable-interval tabular data and commencing searches of tables from the points bounding the previous interpolated value. Odd-order polynomials are proposed to ensure continuity of the interpolated function when the interpolation span consists of the middle portion of the fitting interval. Continuity is ensured because switching of interpolating polynomials occurs at data points that are fit exactly.

#### 2.1.5.6 Statistical Analysis and Digital Filtering

Both corporate Project team members have extensive experience in applying filtering and smoothing techniques such as finite-memory digital filters (minimum



variance, Kalman, weighted least-squares) and polynomial smoothing techniques to experimental data for aeronautical systems. The team is also aware of difficulties that can arise due to low signal-to-noise ratio, data noise (systematic and random), and data editing.

#### 2.1.6 Potential of the System Design to Accommodate Advances in Helicopter Analysis Technology

The inherent modularity and adaptability of the System design provide the growth potential needed to add innovative analysis concepts. Based on the history of new inventions, it is expected that at least one new device, concept, or material will have a strong impact on the helicopter industry during the 15-year cycle of the Long Range System, similar to the way in which composite materials are having several effects at the present time. The use of such unifying and systematic concepts as the finite element concept, advanced dynamic coupling methods, and flexible aerodynamic models allows the System design to adapt to the changing needs of the future without changing its overall structure, thus ensuring a System that can be maintained and extended at minimum cost.

One example of a possible future device is an electrostatic autopilot that has been used on fixed-wing aircraft but, due to the disturbance of the electrostatic field by the rotor, not on helicopters. To analyze such a device, a description of the Earth's basic electrostatic field would be needed. This could be done by using software elements similar to those that describe the aerodynamic field (see Section 2.1.4). A rotor electrostatic formulation could then be defined in a form similar to the rotor wake formulation to calculate the disturbance caused by the rotor to the electrostatic field. Another software element would be the electrostatic autopilot itself, which would make control motions a function of

the electrostatic potential at user-specified node points in a manner very similar to the automatic flight control system that is being used today.

An innovative mechanical device might provide dynamic coupling between any two or more helicopter components. Acceleration coupling could be represented in the mass matrix; velocity coupling, in the damping matrix; and displacement coupling, in the stiffness matrix. Using the finite element technique, unusual dynamic couplings may be achieved and their effects tested and examined analytically rather than by actually building the hardware. The analysis of the effects of such an innovative device would require only the addition of a few new software elements to represent the device.

Also, any device that changes the aerodynamic field or modifies the aerodynamic coupling between the aircraft components can be accounted for by the use of one or the other of the two types of aerodynamic models discussed in Section 2.1.4. Though some new software elements would have to be defined, the two basic types of aerodynamic models would remain unchanged.

Although not required by the Baseline Type A System Specification, the use of the System for optimization studies represents yet another area of potential growth. By using appropriate optimization algorithms, the System could be adapted for such needs. The design of an aircraft with maximum endurance or range, the synthesis of a particular aircraft component with natural frequencies that are designed to avoid resonances with some other components, and optimization of rotor planform and twist in preliminary design are examples of such optimization applications.

The design of the System makes it possible to investigate analysis problems such as those mentioned above as well as other analysis problems by enhancing its capabilities. This may be done without changing the basic structure of the System and at minimum cost.

## 2.2 SYSTEM HIERARCHY

Because of the complexity of the functional requirements placed on the System, the System itself is complex, i.e., composed of many elements. This section defines the terms used to describe the System's elements which are presented in a top-down hierarchical fashion.

The System is made up of a set of modules. Because a module, which is the equivalent of a FORTRAN subprogram, is a relatively small part of the System (a module consists of on the order of 100 executable, high-level language source statements), it is convenient in describing the architecture of the System to give names to particular sets of modules in the System. The names used are complex, subsystem, package, and subpackage. These sets of modules are, in the terminology of set theory, subsets of the System. The terms complex, subsystem, package, subpackage, and module are defined below and summarized in the Glossary for reference.

The System is made up of two complexes: the Operational Complex and the Support Complex. The Operational Complex is that subset of modules of the System used to obtain predictions in the five areas of helicopter analysis previously enumerated. The Support Complex is that subset of modules of the System needed to support the development, test, configuration management, and documentation of the Operational Complex and to support the overall management of the System. The two complexes are mutually exclusive and exhaustive; i.e., a module of the System is an element of the Operational Complex or of the Support Complex but not an element of both, and every module of the System is an element of one of the complexes.

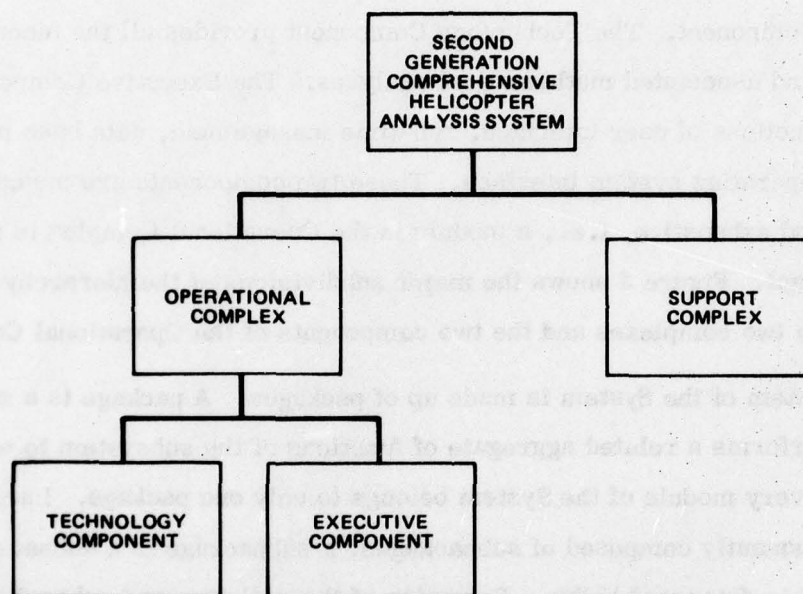
Each complex is made up of subsystems. A subsystem is a set of modules that performs a related aggregate of System functions. For example, one subsystem of the Operational Complex is the Trim Solution Subsystem, which consists of all the modules that are used to find a steady-state solution to the set of equations



represented by the simulation model. A subsystem is not an executable entity. Subsystems are defined for the purpose of relating the software design of the System to the requirements of the Type A System Specification; aiding the presentation of a logical, top-down design; and aiding in configuration management. Every module of the System belongs to one and only one subsystem. In the Operational Complex, a subsystem is characterized as being a technology subsystem or an executive subsystem. The collection of technology subsystems is called the Technology Component, and the collection of executive subsystems is called the Executive Component. The Technology Component provides all the functions of helicopter and associated mathematical analyses. The Executive Component provides the functions of user interface, run-time management, data base management, and operating system interface. These two components are mutually exclusive and exhaustive, i.e., a module in the Operational Complex is in only one component. Figure 3 shows the major subdivisions of the hierarchy of the System: the two complexes and the two components of the Operational Complex.

Each subsystem of the System is made up of packages. A package is a set of modules that performs a related aggregate of functions of the subsystem to which it belongs. Every module of the System belongs to only one package. Large packages are frequently composed of subpackages; a subpackage is a subset of a package and is discussed below. Execution of the packages and subpackages of the Technology Component is controlled by the Run-Time Control Package of the Executive Component. Packages and subpackages may be invoked either as a result of entries in the Sequence Control Table, which is constructed from user-specified data by the Executive Component during the input phase of an analysis run, or as a result of requests from a module of the Technology Component during the processing phase of an analysis run.

Each package may be made up of subpackages. A subpackage is a set of modules that performs a related aggregate of functions of the package to which it belongs. Often when discussing elements of the software system hierarchy, one wishes



**Figure 3. Major Subdivisions of the Software System Hierarchy**

to refer to a collection of modules that are packages or subpackages, but the context calls for a reference to an inspecific level of the System hierarchy. The term software element is used to denote such a collection of modules. Because a package or subpackage may in certain limited instances consist of only one module, a software element can denote a module. A module, the basic element of the System, has the following principal characteristics:

- It performs only one function.
- It has a unique name.
- It constitutes one compilation or assembly.
- An invocable, executable module consists of no more than 100 executable statements. (Not all modules are executable, e.g., a FORTRAN block data subprogram.)
- It has only one point of entry.

Figure 4 shows the hierarchy of subsystem, package, subpackage, and module. The hierarchy of the entire System is obtained by placing Figure 4 under the Technology Component, the Executive Component, and the Support Complex of Figure 3. Rectangles at the subsystem, package, and subpackage levels denote a set of modules. A rectangle at this level does not denote a module driver, nor is the execution sequence of the software implied in any way by the hierarchy diagrams (they are not "who calls who" hierarchy diagrams). In addition, the order in which the rectangles are placed from left to right at any level is quite arbitrary: no execution order or sequencing is implied.

### 2.3 OPERATIONAL COMPLEX

The Operational Complex consists of the Technology Component and the Executive Component. The Technology Component is made up of 10 subsystems, as described in Section 2.3.1. The order of execution of the software elements required to analyze a helicopter model and the data used during an analysis are specified by



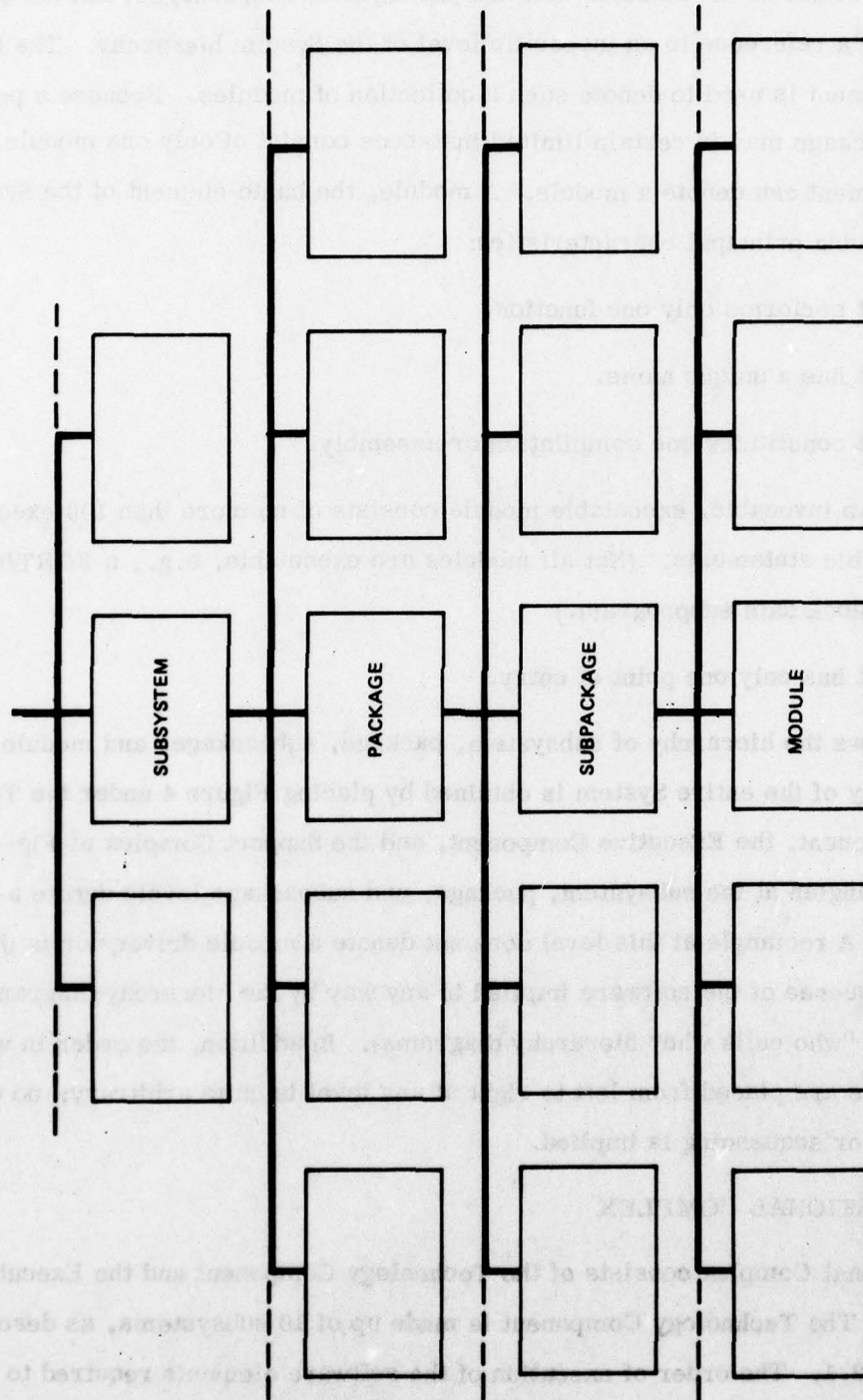


Figure 4. Hierarchy of Subsystem, Package, Subpackage, and Module

the Sequence Control Table, which is constructed by the User Input Package (a package of the Executive Component) from user input for an analysis run.

Communication between and among the software elements listed in the Sequence Control Table is accomplished by allowing a software element to use input data that were calculated by another software element earlier in the execution sequence. Technology Component software elements have the capability to affect the execution sequence dynamically by defining a System Command to be executed immediately by the Executive Component. Thus, a software element is able to issue a command causing the Executive Component to execute a second software element or select additional data from the Run Data Base. Upon completion of the System Command execution, the software element issuing the command continues to operate from the point immediately following the point at which the command was issued.

The Executive Component satisfies the secondary and implied functional requirements necessary for a user-oriented, transportable, easily extendable software system for comprehensive analyses of helicopters. The Executive Component consists of four subsystems, as described in Section 2.3.2: the User Interface Subsystem, the Run-Time Management Subsystem, the Data Base Management Subsystem, and the Operating System Service Subsystem.

The general flow in the Operational Complex of the System is shown in Figure 5. The System operates in three phases in performing an analysis run: the input phase, the processing phase, and the output phase.

During the input phase, the user input is read and interpreted by the User Input Package of the User Interface Subsystem of the Executive Component. Also input to this phase are the Master Command File, the Master Data Base and, optionally, a Restart File.

The Master Command File contains standard System Command subsequences. The User Input Package selects subsequences based on information in user input and

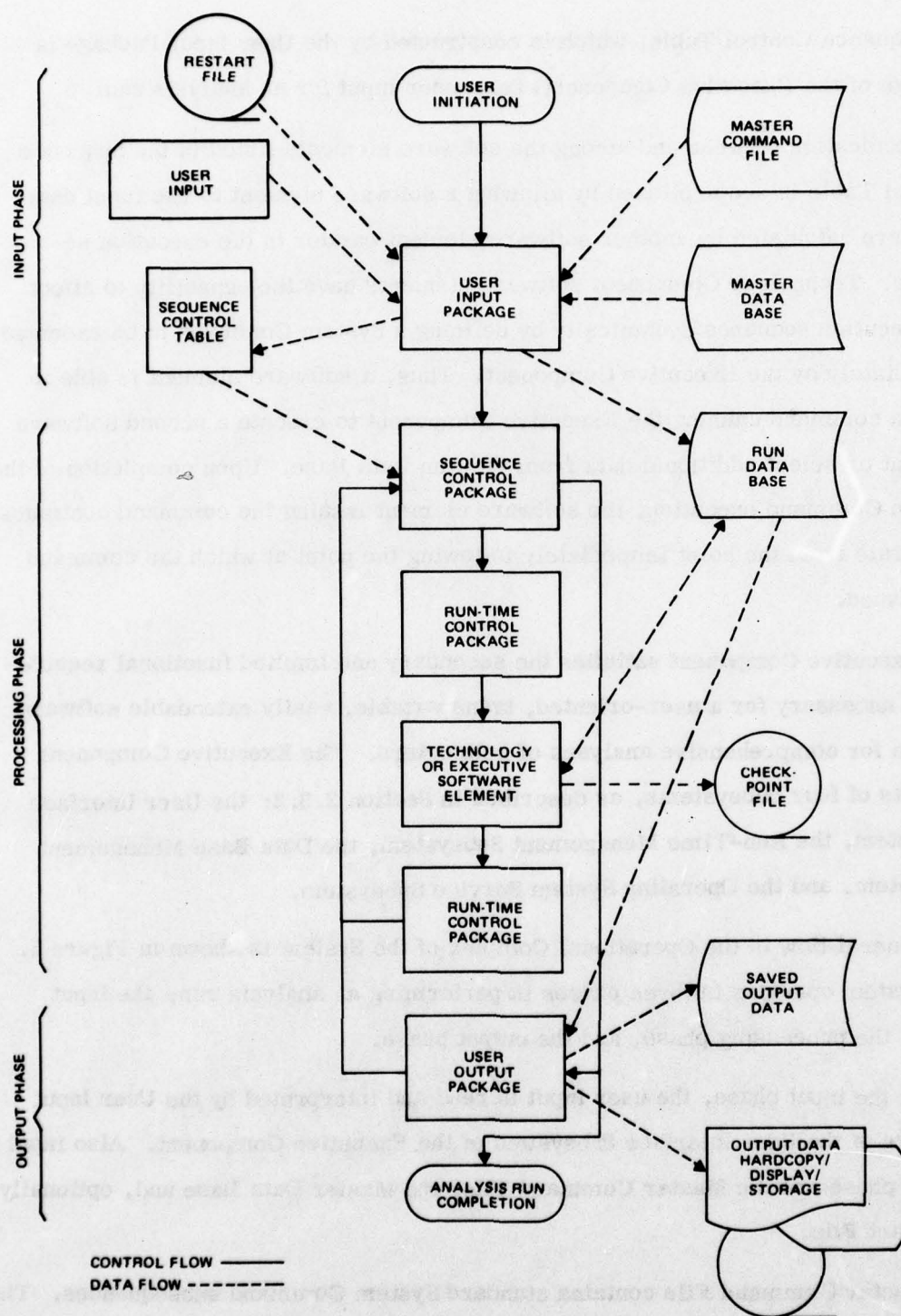


Figure 5. Control and Data Flow for the Operational Complex



constructs from these data the sequence of System Commands for the analysis run. This sequence is stored in internal form in the Sequence Control Table. The Master Data Base contains previously stored descriptions of standard aircraft components and other analysis components; maneuvers, conditions, and operating regimes; and failure/damage effects. The User Input Package selects and modifies data from the Master Data Base based on information in user input and constructs a Run Data Base for an analysis run. The Sequence Control Table defines the sequence of Technology Component software elements and supporting Executive Component software elements to be executed in response to the user input. The Run Data Base contains all data necessary for input to the Technology Component software elements in the sequence. Section 2.5 provides data flow diagrams describing examples of the processing flow and data flow of the sequences and subsequences of System Commands in the Master Command File.

The Restart File is a file that was produced as a Checkpoint File in an earlier analysis run. It contains the results produced at intermediate points in the earlier run. The User Input Package uses the Restart File, the Master Data Base, and the Master Command File, as specified by user input, to establish a Run Data Base and a Sequence Control Table. This capability allows a user to continue an analysis without repeating calculations already performed in an earlier analysis run.

During the processing phase, the Sequence Control Package of the Run-Time Management Subsystem of the Executive Component controls the sequence of technology and executive software elements to be executed as defined in the Sequence Control Table. Before and after each software element is executed, the Run-Time Control Package (of the Run-Time Management Subsystem) is executed to perform housekeeping functions necessary to make input data accessible to the software element and to provide for disposition of its output data. One software element that will be executed when specified by System Command is the Checkpoint Package. The Checkpoint Package saves intermediate results from the Run Data Base

on the Checkpoint File as shown in Figure 5. The Checkpoint File may then be used as a Restart File in a subsequent analysis run to avoid any repetition of already completed calculations. The Sequence Control Package continues to process the entries of the Sequence Control Table until the last table entry has been satisfied.

During the output phase, all data that have been output by the software elements of the Technology Component are printed, plotted, displayed, or recorded on magnetic tape or into a saved output data file for future use by the User Output Package according to directives in the Sequence Control Table. It should be noted that output processing may take place either after the execution of a specific Technology Component software element or after the completion of the sequence of operations for a given analysis run.

The remainder of this section provides a more detailed discussion of the Technology and Executive Components. For each subsystem, the packages of the subsystem are shown on a hierarchy diagram, with each package named in a rectangle. The function performed by the package is given below the rectangle.

### **2.3.1 Technology Component**

The Technology Component is that part of the System which defines the mathematical analysis for the entire simulation. This includes all of the components and couplings described in Section 3.2.2 of the Baseline Type A System Specification. This specification presents several general classes of problems which range from simple to very complex.

The design of the Technology Component was accomplished by dividing the System into major functions and then subdividing these functions to achieve a true top-down structure. The major functions are identified with 10 subsystems as shown in



**Figure 6. No functional overlap exists between or among subsystems. The 10 technology subsystems are**

- 1. Simulation Model Initialization Subsystem**
- 2. Simulation Model Subsystem**
- 3. Trim Solution Subsystem**
- 4. Maneuver Subsystem**
- 5. Stability and Control Subsystem**
- 6. Acoustics Subsystem**
- 7. Aeroelastic Stability Subsystem**
- 8. General Mathematical Operations Subsystem**
- 9. External Models Interface Subsystem**
- 10. Accuracy Assessment Subsystem**

Although 10 subsystems are currently defined, the design of the Technology Component is not rigid; if a new capability is defined in the Type A System Specification, either it is allocated to an existing subsystem or a new subsystem is defined to accommodate it. Subsystems are defined principally for management control and for unity and ease of documentation; because of the ability of the Executive Component to recognize software elements independent of affiliation with a particular subsystem, subsystems may be added, deleted, or reorganized completely without affecting the operation of the System.

Because the System must meet the Government's long-range objectives, the design of the Technology Component must initially be formulated around the solution of the most complex problem. A way must then be found to simplify the formulation so that the solution of simple problems is not penalized by high overhead costs. The mathematical formulation selected is discussed in Section 2.1. The form of the equations is a set of differential equations in time represented by mass, damping, and stiffness matrices. The solution variables of these equations are called the analysis degrees of freedom.



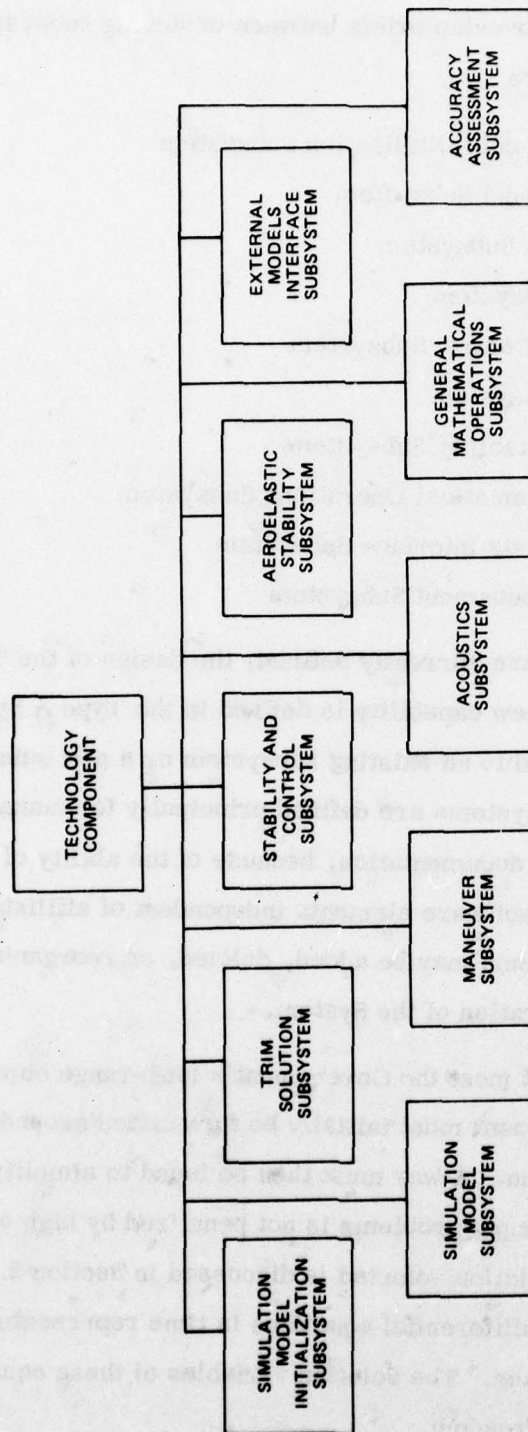


Figure 6. Hierarchical Definition of the Technology Component

The first two subsystems shown in Figure 6, the Simulation Model Initialization Subsystem and the Simulation Model Subsystem, are used to establish and generate the mass, damping, and stiffness matrices and the vector of forcing functions. The Simulation Model Initialization Subsystem establishes coordinate systems, sets internal indexes to indicate how matrices are partitioned, and initializes constant coefficients. The software elements of the Simulation Model Subsystem perform the actual calculations necessary to generate elements of the mass, damping, and stiffness matrices and the vector of forcing functions. The software elements of this subsystem are generally executed many times during the processing of a problem and are invoked by modules in several other subsystems.

The next five subsystems in Figure 6 are directly related to five types of solutions of the equations of motion:

- The Trim Solution Subsystem contains those packages that find a steady-state solution for the simulation model. This solution of the steady-state problem is completed prior to calculating any of the aircraft technical characteristics and is frequently an end in itself.
- The packages of the Maneuver Subsystem are used during a direct timewise integration of the equations of motion. A number of external disturbances to the steady-state condition may be introduced by packages of the Maneuver Subsystem. These include prescribed or automated control motions, gust disturbances, and others, which are described later.
- The packages of the Stability and Control Subsystem calculate frequencies and damping coefficients for the response of the rigid-body motion of the airframe. After performing these calculations, linear transfer functions and airframe rigid-body frequency response may be computed.
- The packages of the Acoustics Subsystem use aerodynamic and rotor wake data obtained during the other processes to calculate the sound

level generated by the rotors and other components. Factors of environment, insulation, and dissipation are then used to calculate the sound level throughout a field of interest.

- The packages of the Aeroelastic Stability Subsystem calculate frequencies and damping coefficients for the aeroelastic rotor/airframe. The packages of this subsystem use one of several methods to find the frequencies and damping of the coupled aeroelastic system.

The eighth subsystem of the Technology Component, the General Mathematical Operations Subsystem, contains packages of a general mathematical nature, such as differential equation solution, eigenvalue solution, and interpolation. The packages in this subsystem are used by software elements of other subsystems of the Technology Component and may be called by the Run-Time Control Package of the Executive Component during the execution of an analysis run.

The ninth subsystem of the Technology Component is the External Models Interface Subsystem. Each package of this subsystem calculates the output required for one external model. This subsystem generates whatever data an external model needs before control is returned to the Executive Component for actual output.

The final subsystem of the Technology Component is the Accuracy Assessment Subsystem. This subsystem calculates the sensitivity factors, standard deviations, and expected value ranges for each of the variables for which the user has requested accuracy assessment data.

Sections 2.3.1.1 through 2.3.1.10 present more detailed discussions of the 10 Technology Component subsystems. As indicated in Section 2.2, each subsystem is divided into packages. Packages in most cases are further subdivided into subpackages. Subpackages and their functions are discussed in Section 3, System Capability. To facilitate the discussion of the subsystems, a hierarchy diagram showing the packages of each subsystem is presented. Each rectangle of the



hierarchy diagram defines the package name, and below the rectangle the function of the package appears.

#### 2.3.1.1 Simulation Model Initialization Subsystem

The packages of the Simulation Model Initialization Subsystem perform initialization functions for constants and aircraft component models that will be used by packages of the Simulation Model Subsystem. The manner in which the subsystem is divided into packages is shown in the hierarchy diagram for the subsystem (Figure 7). Eight packages have been identified to perform initialization and input data checking functions. The analysis definition input data for the simulation includes a description of both the aircraft and its environment. The aircraft description is related to the geometry and structural properties of the aircraft itself. The environment description concerns such analysis components as airmass, wind tunnel, test stand, and ground/deck. The principal subsystem output is the simulation data used during the problem solution.

The Combine Aircraft Components Package performs initialization functions for the parts of the simulation that describe the aircraft. These functions include reasonability analysis of the input data describing the aircraft, initial estimates of solution values, and calculations of constant coefficients for the rotor, airframe, engine/drive system, and control system/pilot models. The aircraft model data that it generates are later used by the Combine Aircraft and Environment Components Package.

The Combine Environment Components Package performs initialization functions for the parts of the simulation comprising the aircraft environment, including reasonability analysis of input data and the setup of the airmass and ground/deck models.

The aircraft and environment models are combined and verified by the Combine Aircraft and Environment Components Package. A reasonability analysis verifies the compatibility of the aircraft and environment models used in the simulation.

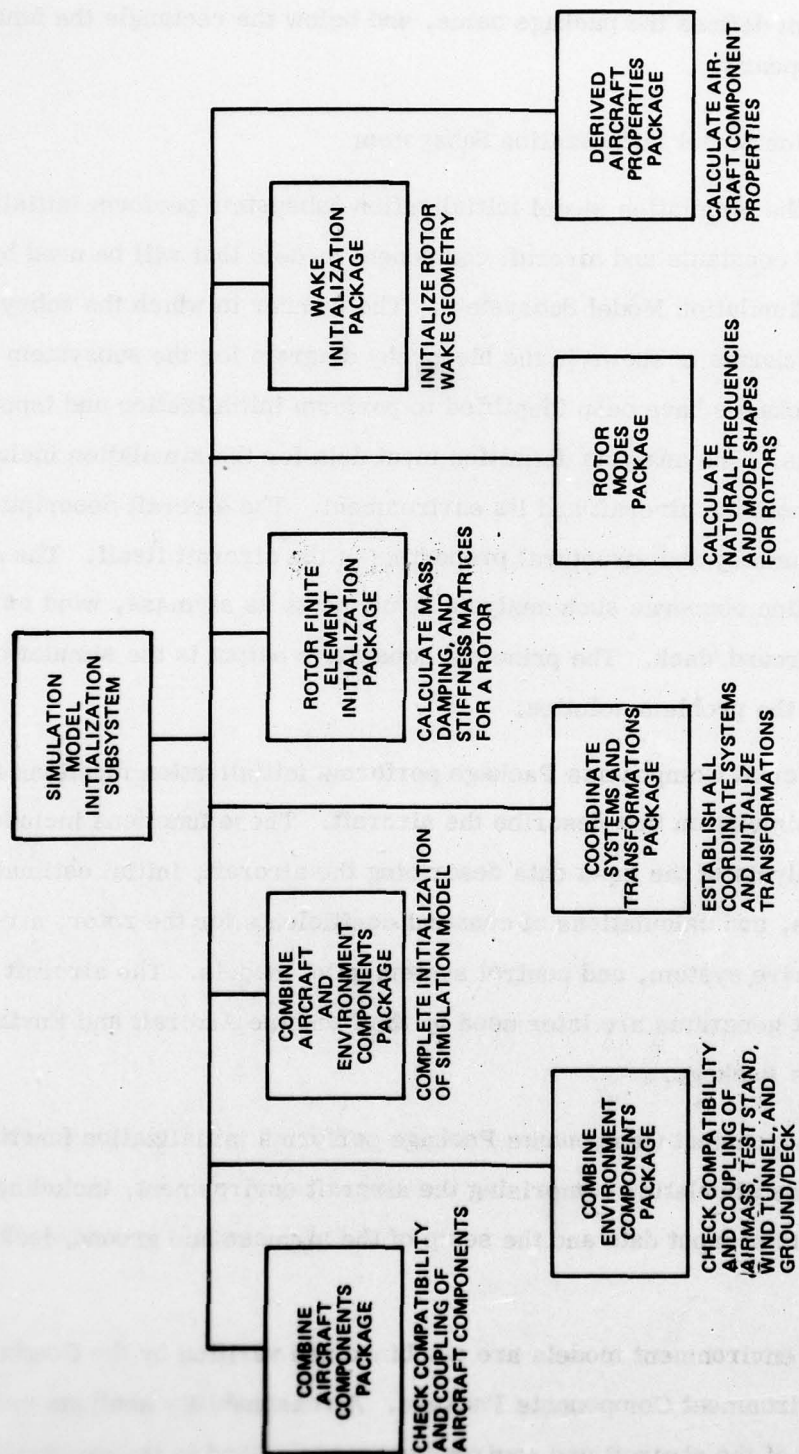


Figure 7. Hierarchical Definition of the Simulation Model Initialization Subsystem

This package also produces data elements in a form ready for use by the Simulation Model Subsystem.

The Coordinate Systems and Transformations Package establishes the number of different coordinate systems required, their initial locations, their initial orientations, and their interrelationships.

The Rotor Finite Element Initialization Package calculates the mass, damping, and stiffness matrices for a finite element rotor analysis. This analysis includes the capability to model redundant load paths.

The Rotor Modes Package computes structural natural frequencies and normalized mode shapes for the rotor. The frequency and mode shape data that are computed may be used in a subsequent simulation analysis, or they may be the primary engineering output required for a particular case.

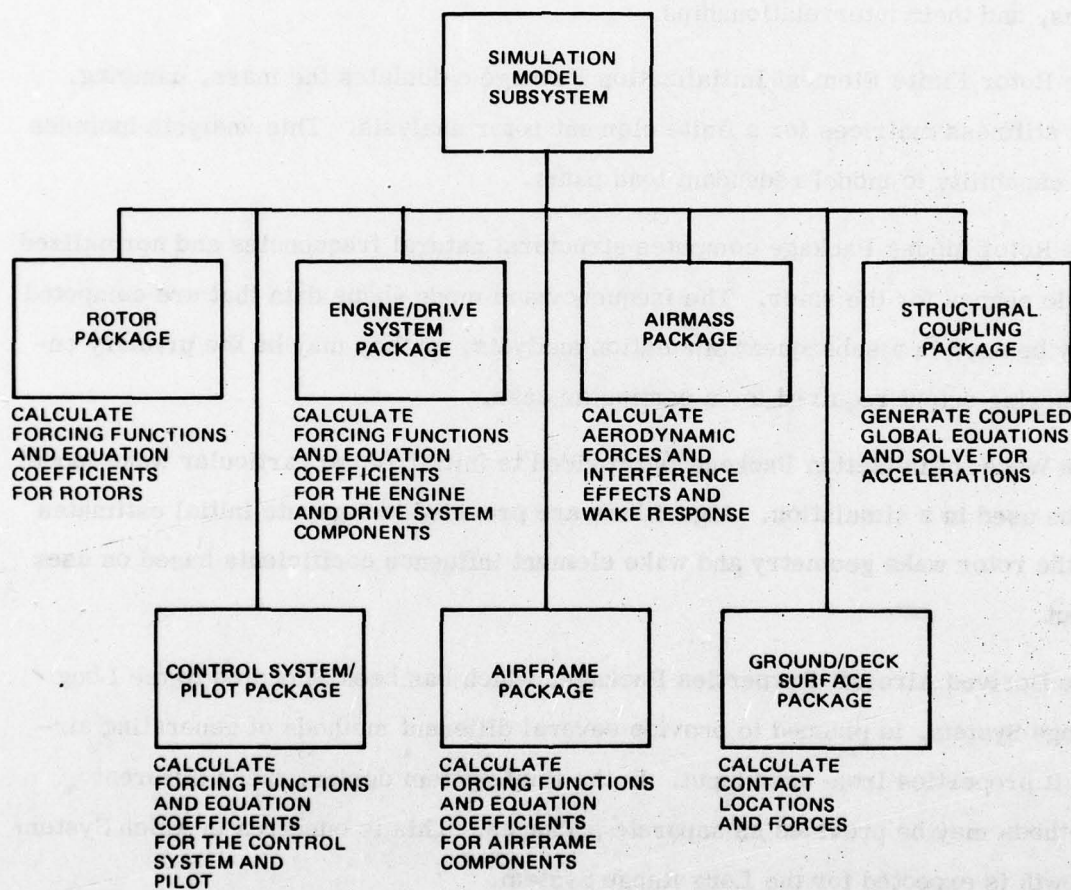
The Wake Initialization Package is provided to initialize the particular wake model to be used in a simulation. Algorithms are provided to compute initial estimates of the rotor wake geometry and wake element influence coefficients based on user input.

The Derived Aircraft Properties Package, which has been allocated to the Long Range System, is planned to provide several different methods of generating aircraft properties from user input. In the final System design, these different methods may be provided as separate packages. This is one area in which System growth is expected for the Long Range System.

#### 2.3.1.2 Simulation Model Subsystem

The Simulation Model Subsystem consists of the software elements that represent the physical character of the configuration. It is composed of seven packages, as shown in the hierarchy diagram for the subsystem (Figure 8). A distinction has been made between packages that model portions of the aircraft and those that





**Figure 8. Hierarchical Definition of the Simulation Model Subsystem**

model the environment. This is a convenient distinction for the purpose of data input and representation of airframe couplings and interactions.

Packages of the subsystem use simulation input and flight conditions data generated by software elements of the Simulation Model Initialization Subsystem to produce (1) forcing functions and coupling coefficients for the equations of motion and (2) distributed aerodynamic loads and vibrations data and the aerodynamic forces and moments on each aircraft component. Only those packages and subpackages needed for analyzing the user's problem are executed during the analysis.

The Rotor Package consists of the subpackages required to model the dynamic behavior of a rotor. The subpackages range in level of complexity from a rotor map table look-up to complete detailed aeroelastic rotor analyses. The indicated methods generally have different levels of complexity. For example, the rigid-blade analysis has one or two degrees of freedom for each blade. In contrast, the advanced rotor models use many node points and analysis degrees of freedom for each blade.

The Control System/Pilot Package includes the aircraft components required to represent the helicopter control system and pilot in a simulation. Subpackages either take the form of complete models of control systems for specific helicopter types (e.g., single-rotor, tandem-rotor, tilt-rotor) or are models of control system components (e.g., mixing boxes, actuators). Models in the latter category may be coupled to form specific control system models.

The characteristics of engines and drive systems are represented by the Engine/Drive System Package. Performance and dynamic response characteristics of various engine types (e.g., reciprocating and turbine) are modeled. Both mechanical and gas-cycle drive systems are represented. Drive system subpackages for flexible and dynamic analyses either take the form of complete drive system models or are models of components (e.g., drive shafts, couplings, gearboxes).

The Airframe Package includes the basic fuselage and a number of appendages, such as rotor pylons, aerodynamic surfaces, landing gear, and external stores. The pylon model is general in nature so that it can also be used to represent the wing of a tilt-rotor aircraft.

The Airmass Package consists of subpackages that compute the location, effects, and interactions of the rotor and wing wakes and evaluate both steady and unsteady aerodynamic forces. Separate subpackages are identified for calculation of aerodynamic coefficients so that they may easily be replaced or upgraded as better aerodynamic analyses are developed. The simplest inflow model is the inclined rotor disk momentum theory model. This model predicts a uniform downwash field on the rotor disk with a superimposed moment of momentum component to give some accounting for the azimuthal variation in blade loading. The more complex class of flow field model is the free-wake vortex element model. Aerodynamic flow fields for bodies and surfaces are calculated using aerodynamic panel methods.

The Ground/Deck Surface Package provides for the aerodynamic effects of a ground plane, a ship deck, or wind tunnel walls. Representations of the dynamic effects of fixed, moving, slanted, bumpy, or elastic ground planes are included for use in taxi, takeoff, and landing studies. A liquid-surface model is implemented as a free-surface representation. Some of these items are beyond the present state of the art, however, and will require additional research work to implement.

The final package of the Simulation Model Subsystem is the Structural Coupling Package. This package provides the capability to couple the structural components that make up the configuration. The uncoupled mass, damping, and stiffness matrices and forcing vectors are transformed into the coupled equations for the entire configuration. The Structural Coupling Package also calculates the global acceleration vector for the entire configuration and derives from it the local acceleration vectors for each component.



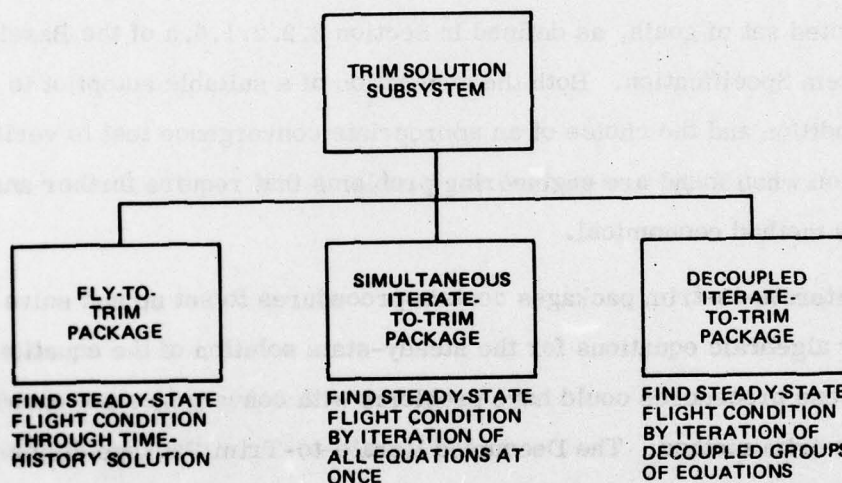
### 2.3.1.3 Trim Solution Subsystem

The Trim Solution Subsystem is used to find a steady-state solution to the set of equations represented by the simulation model. This process is one of finding a set of values for the user-specified trim variables that allows the aircraft to continue the same flight path indefinitely if not disturbed. Three different approaches to the problem are shown in the hierarchy diagram for the subsystem (Figure 9).

The Fly-to-Trim Package performs a simulation of a specialized type of maneuver. This maneuver would be used to find a steady-state condition that satisfies a user-selected set of goals, as defined in Section 3.2.2.1.6.a of the Baseline Type A System Specification. Both the generation of a suitable autopilot to seek the trim condition and the choice of an appropriate convergence test to verify a trim condition when found are engineering problems that require further analysis to make this method economical.

Both of the iterate-to-trim packages contain procedures to set up and solve a set of nonlinear algebraic equations for the steady-state solution of the equations of motion. These procedures could have problems with convergence not encountered by the fly-to-trim method. The Decoupled Iterate-to-Trim Package divides the equations of motion into user-specified groups and converges each group separately in cycles to find the solution. In some cases this eliminates convergence problems experienced with the Simultaneous Iterate-to-Trim Package.

Input and output for all packages are identical. Input consists of simulation input data with coordinate systems and transformations in addition to the specification of the desired trim condition. When the trim condition is found, output includes the steady-state performance data, steady-state loads data (if requested), and the flight condition data with the trim variables determined.



**Figure 9. Hierarchical Definition of the Trim Solution Subsystem**

#### 2.3.1.4 Maneuver Subsystem

The packages of the Maneuver Subsystem simulate all specified types of maneuvers or transient flight conditions. The maneuvers may include prescribed control motions, gust disturbances, landings, the following of a specified flight path, component failure, aircraft damage, or other options, as described in paragraphs b, c, and d of Section 3.2.2.1.6 and in Section 3.2.2.1.7 of the Baseline Type A System Specification. Figure 10 shows that the subsystem is divided into four packages.

The Prescribed Control Motions Package converts the user's input for control motions into functions of time for control positions. The controls which may be moved include all of the primary and secondary controls.

The Prescribed Aircraft Response Package calculates motions of the primary controls during a time-history solution to attempt to achieve a user-specified response of some aircraft motion. The most common example of this function is the following of a prescribed time history of vertical load factor without excessive rolling or yawing motions.

All types of airflow disturbances are calculated by the Gust Response Package. This package calculates the disturbance in the fluid flow field as a result of gusts, trailing vortices, or weapons blast. The effects are calculated at every aerodynamic node point of the configuration.

The Initiate Failure/Damage Effects Package activates the failure or damage effects for which the user has made provision. These may be activated at a certain time or by the impact of two components such as a rotor and fuselage. The representation of the damage itself is accomplished in the subpackages of the Simulation Model Subsystem.



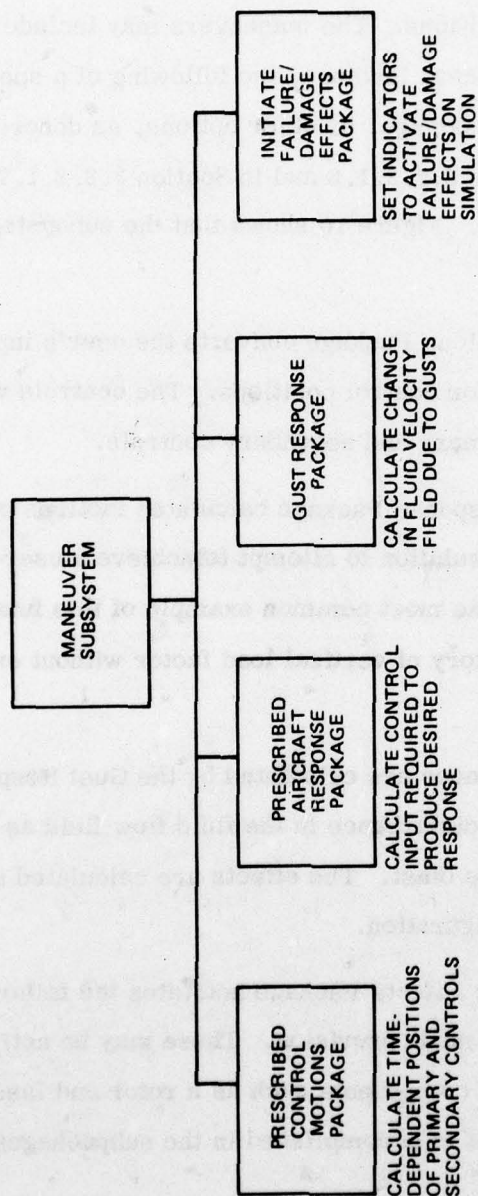


Figure 10. Hierarchical Definition of the Maneuver Subsystem

#### 2.3.1.5 Stability and Control Subsystem

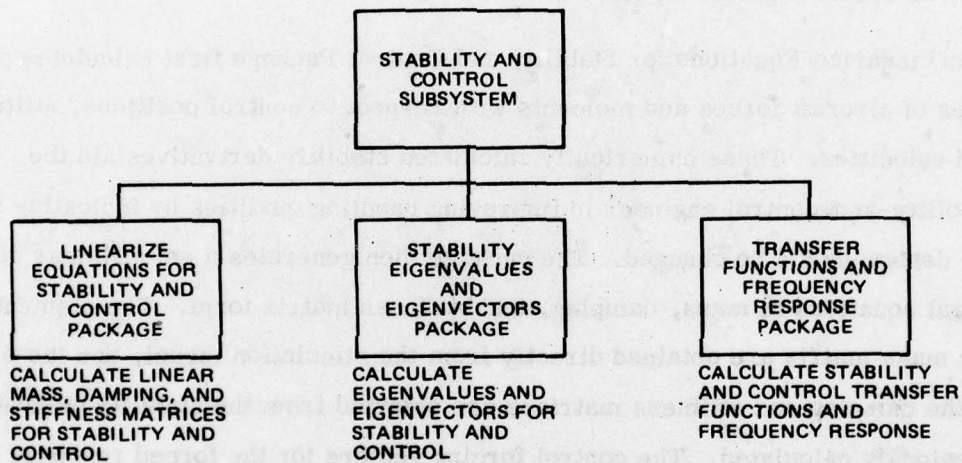
The packages of the Stability and Control Subsystem evaluate the specialized data required for the stability and control part of helicopter analysis. The subsystem is divided into three packages, as shown in the hierarchy diagram (Figure 11).

These three packages constitute a sequence of processes used to produce the stability and control data. These data are the stability eigenvalues and eigenvectors, transfer functions, and frequency responses. The stability derivatives may also be useful engineering output.

The Linearize Equations for Stability and Control Package first calculates derivatives of aircraft forces and moments with respect to control positions, attitudes, and velocities. These numerically calculated stability derivatives aid the stability-and-control engineer in improving handling qualities by indicating where the design should be changed. The package then generates a set of linear differential equations in mass, damping, and stiffness matrix form. The elements of the mass matrix are obtained directly from the simulation model, and the elements of the damping and stiffness matrices are obtained from the stability derivatives previously calculated. The control forcing vectors for the forced response calculations also use the stability derivatives.

The Stability Eigenvalues and Eigenvectors Package calculates the actual stability eigenvalues for the aircraft with the controls fixed and calculates the degree of control that can be achieved when the controls are moved.

The Transfer Functions and Frequency Response Package makes use of the stability eigenvalues and eigenvectors data. The transfer functions are really an expression of the combined stability roots in fractional form to give the response of some aircraft degree of freedom to a movement of one of the controls. The frequency response data give the amplitude of motion for some aircraft degree of freedom in response to harmonic control motions at various frequencies.



**Figure 11. Hierarchical Definition of the Stability and Control Subsystem**



#### 2.3.1.6 Acoustics Subsystem

The Acoustics Subsystem includes packages for the empirical and theoretical analyses to define the acoustic emissions of the aircraft. The hierarchy diagram of the subsystem (Figure 12) shows each of the packages representing one of the contributing factors to the overall sound level.

The Rotor Sound Package contains subpackages for the sound of the main, tail, and tilt rotors.

The Engine Sound Package contains representations for two basic sources: inlet and exhaust sound. Installation effects, engine type, duct and exhaust treatments, and governor characteristics are taken into account.

The Gearbox Sound Package includes a number of subpackages: main transmission, engine gearbox, accessory gearing, tail rotor gearboxes, and interconnect gearing and shafting. Type of gearing, house/case sound transmissibility, isolation, and installation effects are considered.

The Accessories Sound Package contains several subpackages representing oil cooler fan(s), hydraulic system pumps, bypass valves, ECU, APU, ventilation fans and blowers, generator, alternator, and electrical/avionics equipment. Accessory type, operating mode, and installation effects are taken into account.

The Sound Propagation Package provides a means to trace the sound from the sources represented by the other four packages to an observer. It will account for sound dissipation in the air, through and around materials and structures, and the mixed airflows.

Input to the Acoustics Subsystem includes the aircraft position relative to the observer, free-field or restricted environment, flight conditions, rotor orientation, engine/drive system/accessory location and orientation, airframe interference, sound transmission loss, and the airmass properties. The processes encompass the major external and internal sound sources and their propagation. Output

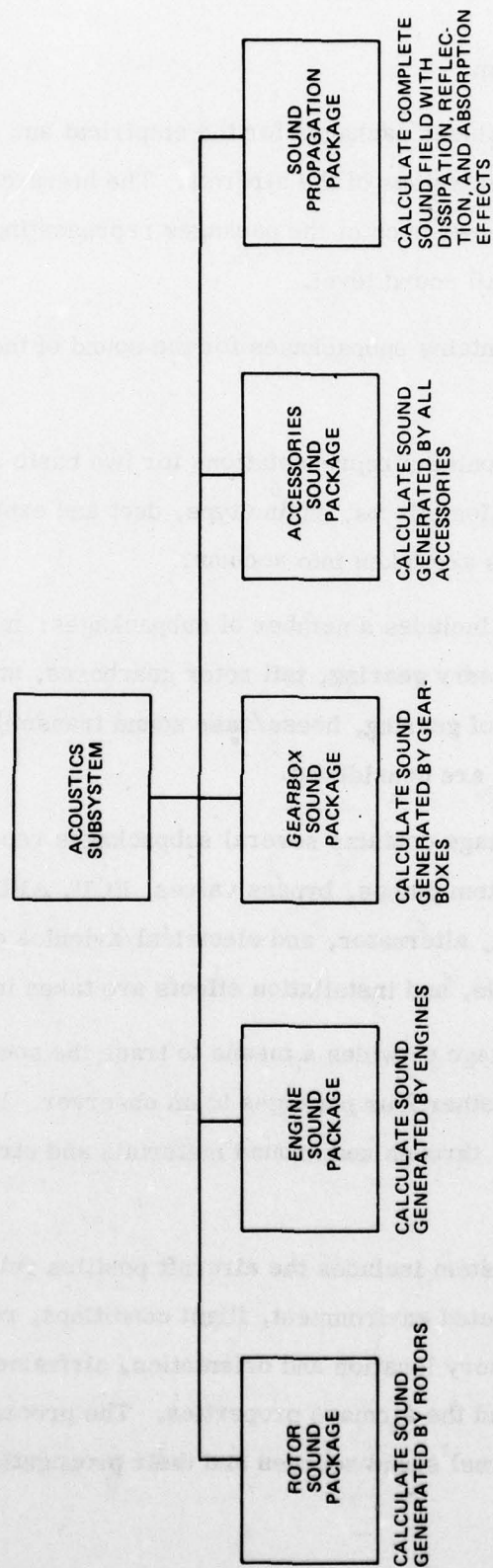


Figure 12. Hierarchical Definition of the Acoustics Subsystem

includes the near-field, far-field, and internal sound levels and the sound signature at detection ranges.

#### 2.3.1.7 Aeroelastic Stability Subsystem

The packages of the Aeroelastic Stability Subsystem calculate the aeroelastic stability characteristics of the aircraft. Stability characteristics are considered aeroelastic when strongly influenced by the elastic deformation of the rotor, the airframe, or other components of the aircraft. The hierarchy diagram of the subsystem (Figure 13) indicates three approaches to the calculation of aeroelastic stability. These approaches allow the user to select a method according to the complexity of the simulation model and the nature of instability to be located. Thus different methods may be used for flutter calculations and for ground resonance.

The Linear Aeroelastic Stability Analysis Package uses linear differential equations with constant coefficients and is thus similar to the Stability and Control Subsystem. The necessary derivatives are first generated so that the equations of motion may be formulated. The eigenvalue problem is then solved for the frequencies, damping coefficients, and aeroelastic mode shapes.

The Floquet Analysis Package uses an analysis that was derived for linear differential equations with periodic coefficients. The damping coefficients are determined from the eigenvalues of the Floquet transition matrix. Several methods are currently available to calculate the transition matrix.

The Aeroelastic Stability Postprocessing Package generates a time history of the parameters of interest in the stability analysis. One or more numerical techniques may at that point be applied to the time-history data in order to obtain the frequencies and damping coefficients.

All three packages have the same input and the same output. Input required includes simulation model data and flight conditions. Output includes the frequencies and damping coefficients that define the aeroelastic stability.



AD-A063 921

COMPUTER SCIENCES CORP SILVER SPRING MD SYSTEM SCIEN--ETC F/G 1/3  
THE PREDESIGN PHASE OF THE SECOND-GENERATION COMPREHENSIVE HELI--ETC(U)  
OCT 78

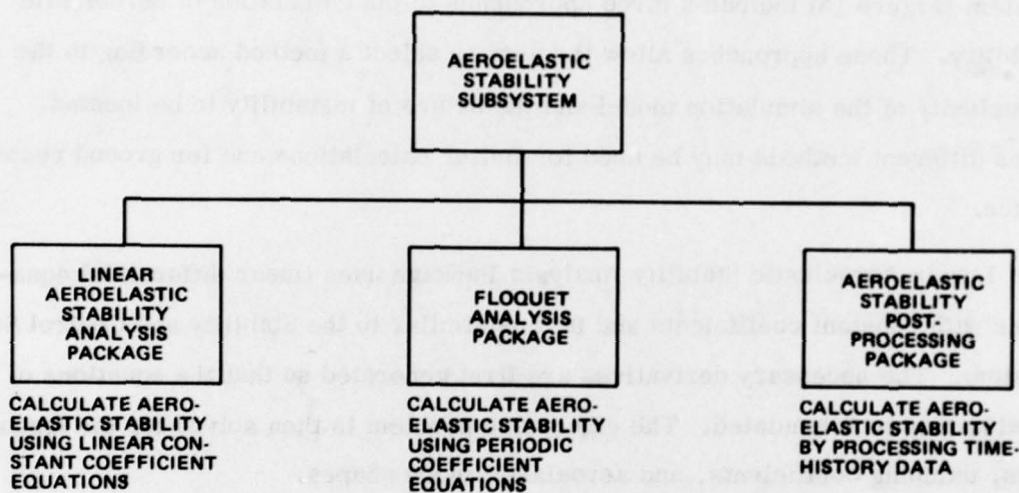
UNCLASSIFIED

USARTL-TR-78-41

NL

2 OF 5  
AD  
A083921





**Figure 13. Hierarchical Definition of the Aeroelastic Stability Subsystem**

#### 2.3.1.8 General Mathematical Operations Subsystem

The General Mathematical Operations Subsystem consists of a set of general-purpose mathematical analysis packages to be used as utilities by other software elements of the System. The packages of the General Mathematical Operations Subsystem are the type of procedures that might be extracted from NASTRAN or some other source. Packages in this subsystem are identified in Figure 14.

#### 2.3.1.9 External Models Interface Subsystem

To satisfy the requirements of the Baseline Type A System Specification for the external models functional capability, the External Models Interface Subsystem is included in the Technology Component. The capability for all external models is included in this subsystem. The hierarchy diagram for the subsystem (Figure 15) indicates the potential for adding packages as external models are defined.

#### 2.3.1.10 Accuracy Assessment Subsystem

The Accuracy Assessment Subsystem consists of the packages necessary to provide the accuracy assessment functional capability of the Baseline Type A System Specification. The four packages are shown in the hierarchy diagram for this subsystem, Figure 16.

The Set Up Accuracy Assessment Cases Package defines a series of cases to be run with perturbations to the input evaluation variables. The equivalent of user input for a series of cases is the output of this package. Following the execution of this package, the series of cases is run using a single System Command Sequence; the resulting output is stored in the Accuracy File.

The Compute Sensitivity Factors Package calculates partial derivatives of each of the output evaluation variables with respect to each of the input evaluation variables. These partial derivatives reflect the sensitivity of the output to error content in the input.



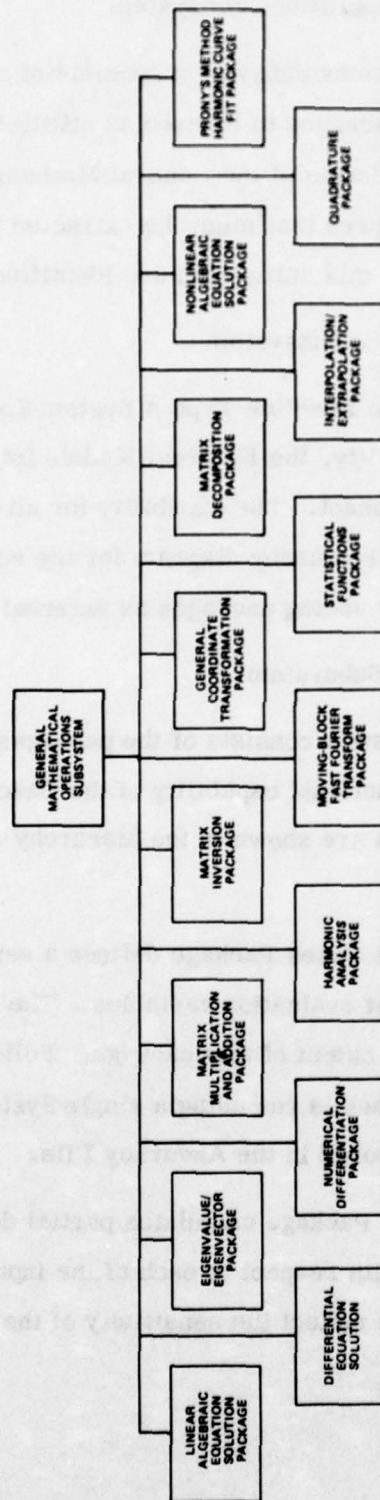
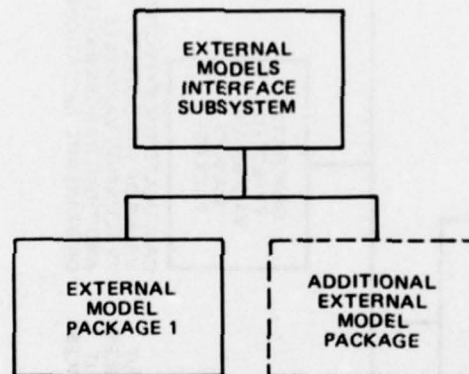


Figure 14. Hierarchical Definition of the General Mathematical Operations Subsystem



**Figure 15. Hierarchical Definition of the External Models Interface Subsystem**

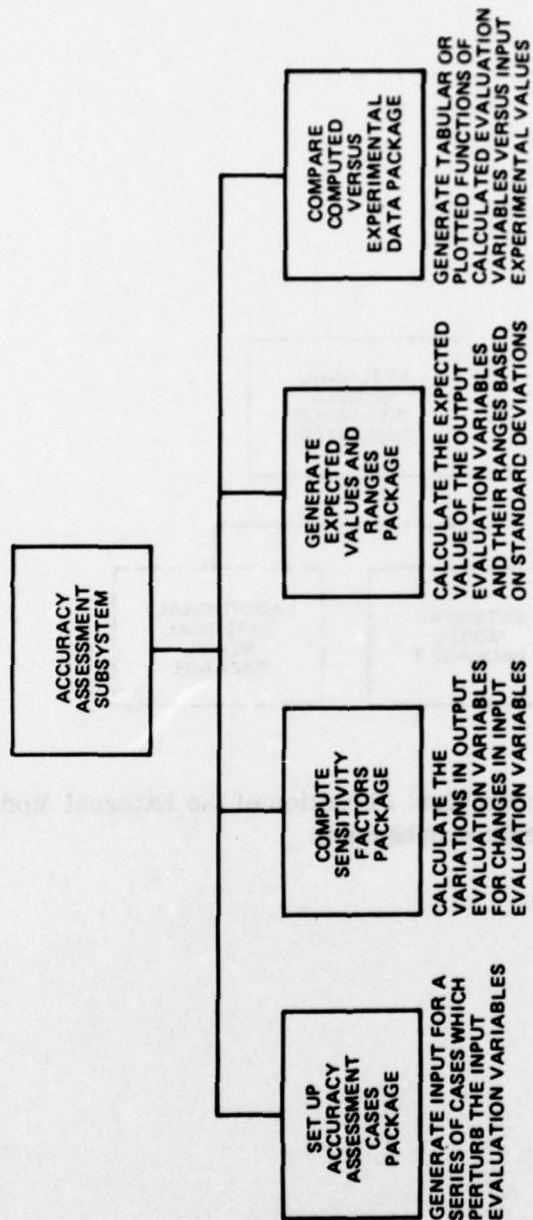


Figure 16. Hierarchical Definition of the Accuracy Assessment Subsystem



The Generate Expected Values and Ranges Package uses sensitivity factors to find standard deviations for the output variables as a basis for expected values and reasonable ranges for experimental error.

The Compare Computed Versus Experimental Data Package performs several different types of comparisons to help the user evaluate the validity and accuracy of the System results.

### **2.3.2 Executive Component**

The Executive Component of the Operational Complex provides the capability to (1) read and verify data at the beginning of an analysis run; (2) construct the Sequence Control Table, which defines the sequence of software elements of both the Technology Component and the Executive Component to be executed during an analysis run; (3) execute software elements of both the Technology Component and the Executive Component; (4) supply the specified input data to the software elements; (5) properly format and route the output data; and (6) perform other services (e.g., provide diagnostic messages) required to keep the Operational Complex functioning smoothly and efficiently. A small portion of the Executive Component is permanently resident in main memory while the Operational Complex is executing. This is primarily the portion responsible for loading load-modules (a load-module is an executable software entity comprised of one or more software elements). The Executive Component is made up of the User Interface Subsystem, the Run-Time Management Subsystem, the Data Base Management Subsystem, and the Operating System Service Subsystem, as shown in the hierarchy diagram (Figure 17). These subsystems are described in Sections 2.3.2.1 through 2.3.2.4, respectively.

#### **2.3.2.1 User Interface Subsystem**

The User Interface Subsystem receives and interprets all direct user input and selects and formats all direct user output whether in batch mode or interactive mode. It also provides the interactive user with meaningful and helpful responses

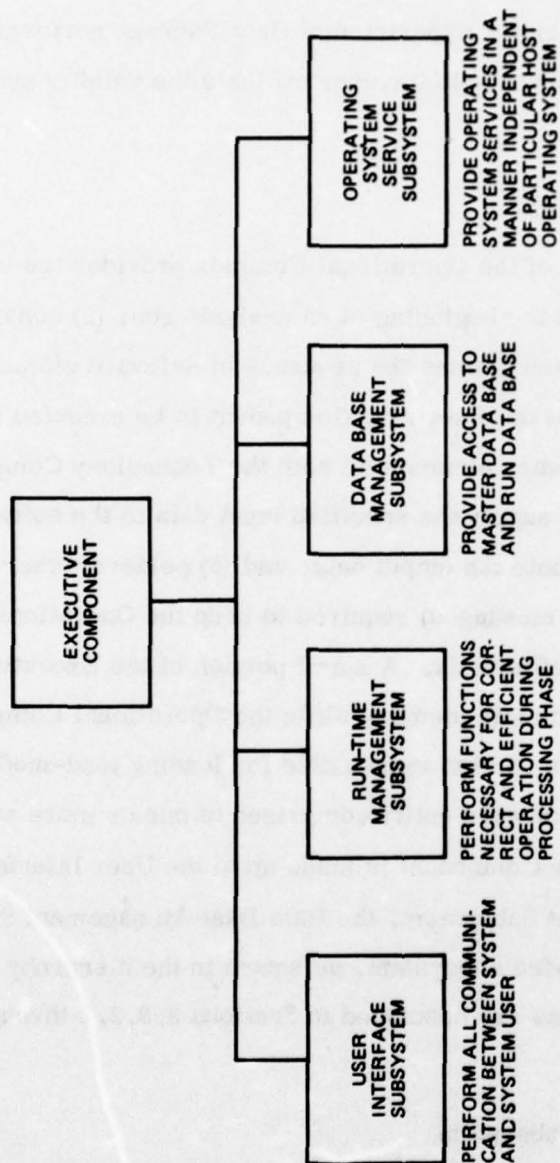


Figure 17. Hierarchical Definition of the Executive Component

during interaction with the System. The User Interface Subsystem consists of the three packages shown in the hierarchy diagram (Figure 18). The packages of the User Interface Subsystem are described below.

The User Input Package reads the file containing the input for an analysis run and generates the Run Data Base and the Sequence Control Table. The Run Data Base is generated by first locating in the Master Data Base the descriptions of aircraft components and other analysis components; maneuvers, conditions, and operating regimes; and failure/damage effects specified for the analysis. The data extracted from the Master Data Base is then augmented or replaced by data specified in user input. (The Master Data Base is generated by the Data Base Support Package in the Support Complex; see Section 2.4 for a discussion of the Support Complex.) The Sequence Control Table is generated by selecting System Command subsequences from the Master Command File, modifying them as required by information in the user input, and concatenating them into the correct System Command Sequence to perform the analysis specified in the user input. (The Master Command File is generated by the Data Base Support Package in the Support Complex.) If the analysis run is a restart, the User Input Package also uses the Restart File created previously (when data from an analysis run was checkpointed) to establish the Run Data Base and the Sequence Control Table. The User Input Package checks the user's input for consistency and correctness as it generates the Run Data Base and the Sequence Control Table, reporting in diagnostic messages any possible errors it detects.

The Interactive Terminal Package, in conjunction with the Interactive Terminal Management Package of the Operating System Service Subsystem, provides all direct interaction with a user at an interactive terminal. The Interactive Terminal Package supplies responses to user input statements, including tutorial information designed to help the interactive user of the System. If any user requests at the terminal require the use of other parts of the system, the Interactive Terminal Package issues a sequence of System Commands for execution just as the User



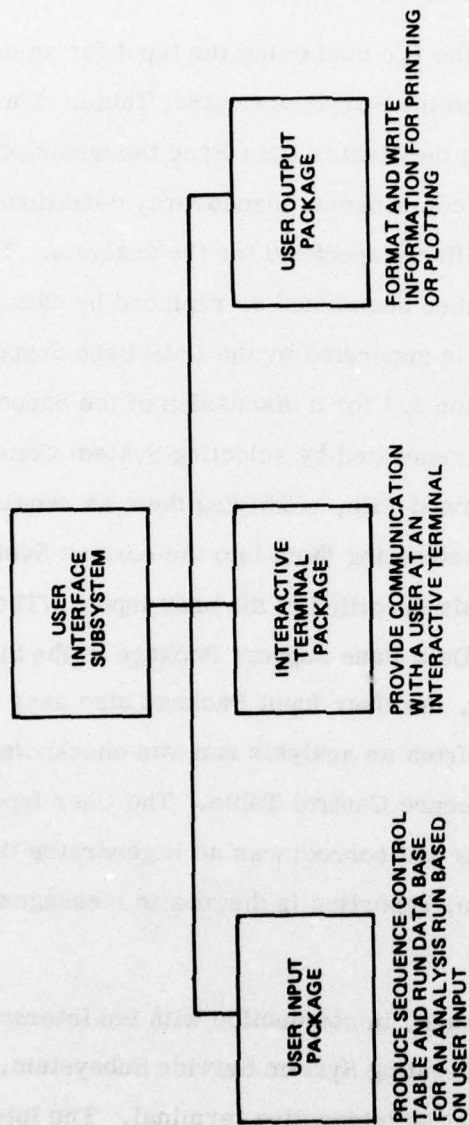


Figure 18. Hierarchical Definition of the User Interface Subsystem

Input Package does. For example, if a user at an interactive graphics terminal requests a plot of a set of data, the Interactive Terminal Packages issue System Commands that cause a plot subpackage from the User Output Package to be executed. The resulting plot is then displayed on the terminal by the Interactive Terminal Package.

The User Output Package prepares all data that are to be directly output to a user. This preparation includes formatting data to be printed and preparing device-independent plotter instructions for data to be plotted. Input to the User Output Package, in addition to the data to be output, is the specification of the format in which the data are to be presented to the user. Subpackages of the User Output Package (e.g., one which produces plots) may be scheduled for execution at any point in the System Command Sequence for output of intermediate results as desired by the user.

#### 2.3.2.2 Run-Time Management Subsystem

The Run-Time Management Subsystem supervises the operations that take place during the processing phase of an analysis run. This subsystem is divided into four packages, as shown in Figure 19.

The Sequence Control Package is responsible for the execution in the proper order of System Commands from the Sequence Control Table. The Sequence Control Table is produced by the User Input Package of the User Interface Subsystem. The System Commands in the Sequence Control Table are of two types:

- An execution command names a software element to be executed and gives a set of parameters for the software element. The Run-Time Control Package is called to execute this type of command. The large majority of the software elements to be executed are software elements of the Technology Component. Some, however, will be from the Executive Component. Examples of Executive Component software elements are the execution of the Checkpoint Package to take a checkpoint at

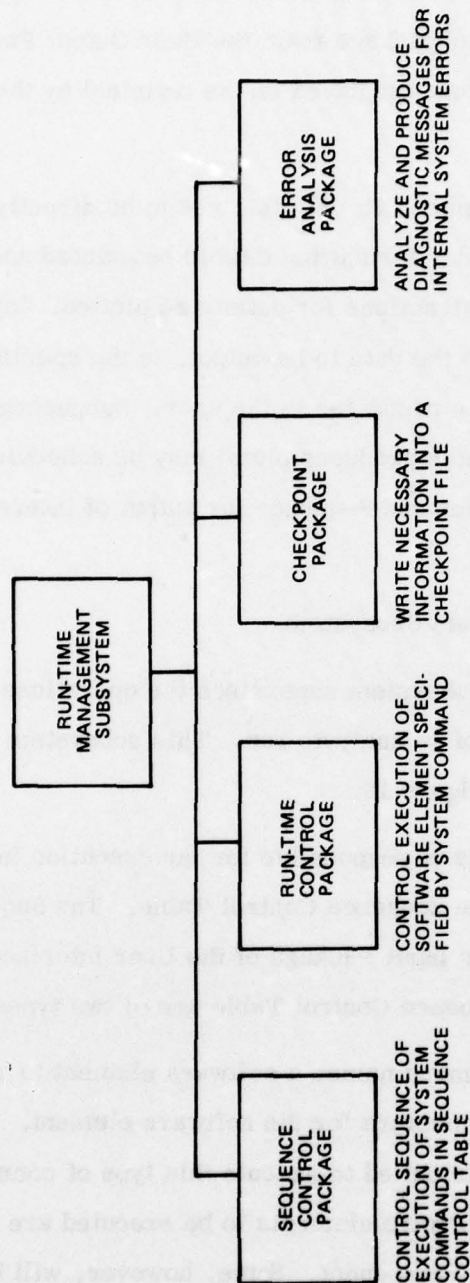


Figure 19. Hierarchical Definition of the Run-Time Management Subsystem



any point in the System Command Sequence or the execution of one of the subpackages of the User Output Package to print or plot any combination of intermediate results in any available format.

- A sequence control command causes the Sequence Control Package to begin command interpretation at a new place in the Sequence Control Table. These commands may be conditional, in which case the Sequence Control Package evaluates the condition to determine whether or not the change of sequence is actually to occur.

The Run-Time Control Package provides the capability to (1) load a load-module for execution, (2) verify that the required data are available to software elements and satisfy feasibility criteria, and (3) begin execution of software elements. In the process of performing this task, the Run-Time Control Package increments a counter associated with the software element being called and updates other performance-measurement statistics as required. The Run-Time Control Package also interrogates diagnostic indicators returned after a software element has executed and produces appropriate diagnostic messages.

The Checkpoint Package saves on a Checkpoint File all of the information required to define the state of an analysis run so that the information can be used in a subsequent run without reexecuting the software elements which create the information. The Checkpoint Package specifically saves on a Checkpoint File the contents of the sets of the Run Data Base indicated in a Checkpoint System Command. The information on the Checkpoint File is organized so that the analysis run can be restarted immediately following any completed Checkpoint System Command. Upon restart, the user may modify either (1) the user input data required by any System Command following the point selected for restart, (2) the System Commands subsequent to the selected restart point, (3) output reporting anywhere in the System Command Sequence, or (4) any combination of the preceding three possibilities.

The Internal System Error Analysis Package generates information needed to identify the cause of internal System errors (i.e., errors not in user input). Upon detection of a possible software error, the type of error detected is reported, the location in the software where the error surfaced is identified, the file (if appropriate) on which the error occurred is specified, and, depending on the user's option, a dump of either the software element wherein the error surfaced or the entire memory region assigned to the Operational Complex is generated. Regardless of the error detected, reports describing current memory utilization, file activity since run initiation, and execution statistics since the start of the run are also produced.

#### 2.3.2.3 Data Base Management Subsystem

The Data Base Management Subsystem provides the capability to access the data in the Master Data Base and the Run Data Base. In order that the System be modular and to allow for the development of Technology Component software elements over an extended period of time, the data required for, and generated by, a software element must be identified other than by means of its location on disk or its relative location in a record. The control of access to such data, stored either in the Master Data Base or the Run Data Base, is provided by the Data Base Management Subsystem. The Data Base Management Subsystem uses a dictionary which contains the name and location of data-items, arrays of data-items, and sets in the Master Data Base or Run Data Base. A discussion of data-items, arrays, and sets can be found in Section 4.2. Software elements of the Technology Component have knowledge only of the names of these data; access to the dictionary and hence to the location of the data is restricted to software elements of the Data Base Management System. If the format of the Master Data Base is changed during the life of the System, only the dictionary need be changed; the software elements that use the data can remain intact, thus avoiding costly maintenance effort. The packages of the Data Base Management Subsystem that allow access to the Master Data Base and the Run Data Base and that allow update of the Run

Data Base are shown in Figure 20. The process of creating and updating the dictionary and the Master Data Base is done by the Data Base Support Package in the Support Complex of the System (see Section 2.4).

The Data Storage Package stores the data supplied to it in locations in the Run Data Base associated with the names of the data supplied to it. The Data Storage Package stores individual data-items, arrays of data-items, or sets.

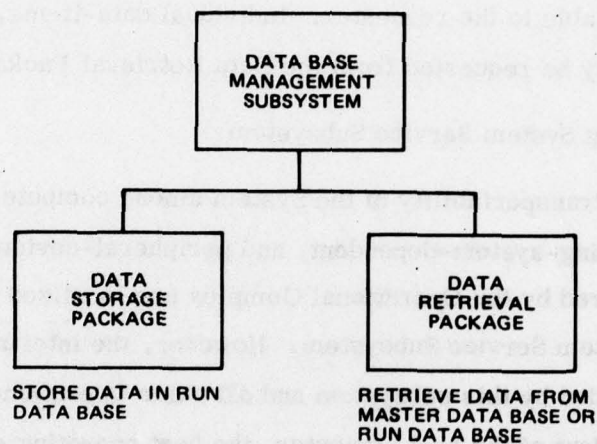
The Data Retrieval Package uses the dictionary to locate requested data in either the Master Data Base or the Run Data Base, reads the data from the data base, and makes it available to the requester. Individual data-items, arrays of data-items, or sets may be requested from the Data Retrieval Package.

#### 2.3.2.4 Operating System Service Subsystem

To maximize the transportability of the System among computers, all computer-dependent, operating-system-dependent, and peripheral-device-dependent capabilities required by the Operational Complex are localized in one subsystem: the Operating System Service Subsystem. However, the interfaces between the capabilities provided by this subsystem and all other Operational Complex subsystems are independent of the host computer, the host operating system, and the peripheral devices of the host computers. This host-computer-independent approach therefore preserves the transportability characteristic of all other Operational Complex subsystems.

To minimize development costs associated with the Operating System Services Subsystem, all subsystem capabilities maximize the use of services provided by the host operating system and minimize the use of software expressly developed or purchased for the System. To simplify the use of the capabilities provided by this subsystem, all services are provided in the form of invocable modules with computer-independent calling sequences. This direct-linkage approach eliminates the introduction of any software linkage overhead which would be incurred if the linkage were indirect.





**Figure 20. Hierarchical Definition of the Data Base Management Subsystem**

The capabilities provided by the Operating System Service Subsystem correlate with services commonly provided by modern operating systems. As presented in Figure 21, five sets of operating-system-like services (capabilities) are provided by this subsystem. These five sets of services directly correlate to five packages: the File Management Package, the Interactive Terminal Management Package, the Program Management Package, the Storage Management Package, and the Cost Assessment and Diagnostic Services Package. Because maximum use is made of host computer operating system services, each of the five packages is composed of one subpackage for each host computer family. (The Host 1 computer family consists of IBM S/370s and S/360s; the Host 2 computer family consists of CDC 6000 and CYBER series computers.) However, there is only one such subpackage of each package on any given host computer: the subpackage which uses the services provided by the host operating system.

The File Management Package provides the following physical input/output services (e.g., read, write, position):

1. Prepare a file for input/output
2. Find a specified record
3. Read a record
4. Add a record
5. Replace a record
6. Update a record
7. Delete a record
8. Terminate file input/output
9. Position a file for subsequent input/output

The Interactive Terminal Management Package provides the input/output interface between the System and the physical interactive or graphics terminals employed

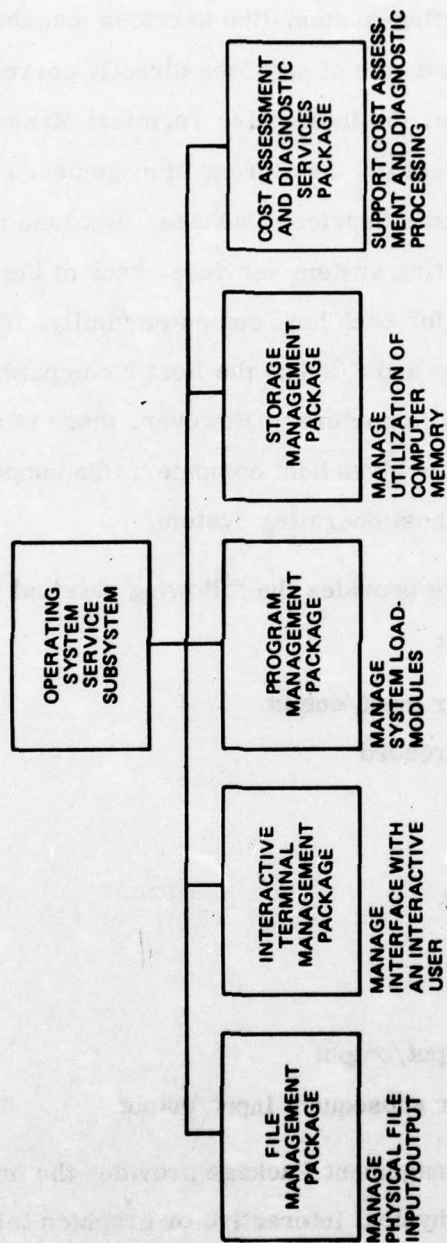


Figure 21. Hierarchical Definition of the Operating System Service Subsystem



by the user to interact with the System. Specific capabilities provided by this package include the following:

1. Accept a message entered by the user at an interactive terminal
2. Notify the Interactive Terminal Package whenever a user message is received
3. Translate user-entered messages into a device-independent format for subsequent processing by the System
4. Reformat System-generated displays (textual and/or graphical) into a device-dependent format for subsequent presentation on the user terminal

The Program Management Package provides the following services for managing load-modules:

1. Bring a load-module into memory if not already in memory
2. Execute a load-module (this is, in effect, a subroutine call)
3. Transfer control to a load-module (when a module  $m_1$  transfers control to a module  $m_2$ ,  $m_2$  returns to the invoker of  $m_1$ , rather than to  $m_1$ )
4. Suspend or terminated load-module execution
5. Delete a load-module from memory if there are no outstanding requests to execute it

The Storage Management Package provides the following services for managing the utilization of computer memory assigned to the Operational Complex:

1. Acquire a block of memory
2. Release a block of acquired memory

The Cost Assessment and Diagnostic Services Package provides the following services needed to support cost assessment processing and diagnostic processing:

1. Generate meaningful dumps of specified memory locations
2. Provide current date (Julian and Gregorian)
3. Provide current clock time (seconds since midnight)
4. Route a message to the computer operator
5. Provide the estimated cost of a run

The estimation of run cost is based upon a computer-independent cost algorithm. However, the algorithm does allow the specification of factors to account for differing performance characteristics of host computers.

#### 2.4 SUPPORT COMPLEX

Successful development of the Second Generation Comprehensive Helicopter Analysis System is dependent upon a comprehensive plan for developing and maintaining the System. A principal result of the Predesign Phase contract is such a plan, a summary of which is presented in Section 5. However, for such a sophisticated System, the mere existence of a comprehensive plan for its development and maintenance will not ensure its success. Automated support of the development and maintenance efforts is also needed if the high quality standards specified in Sections 3.4 and 4 of the Baseline Type A System Specification are to be realized. This automated support is provided in the Support Complex.

The automated support tools defined in the Support Complex are also designed to aid the methods developer in experimenting with and developing new or advanced analysis capabilities. For this reason, the Support Complex is intended to be an integral part of the Long Range System, i.e., the Support Complex contains automated tools to minimize the costs of development during the Development Phase, to control the System configuration during the Maintenance Phase, and to aid the methods developer in experimenting with or developing new or advanced analysis capabilities throughout the life cycle of the System.

Figure 22 depicts the four categories of automated support necessary to ensure a successful System development and maintenance effort. These four support categories correspond to the four Support Complex subsystems: the Development Support Subsystem, the Testing Support Subsystem, the Configuration Management Support Subsystem, and the Documentation Support Subsystem.

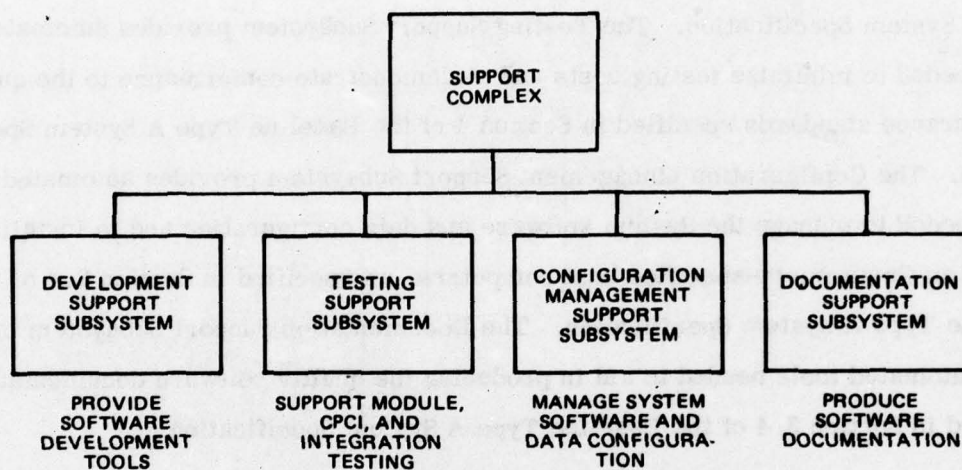
The Development Support Subsystem provides automated tools needed to generate the System software in a cost-effective manner and to ensure conformance of the software to the high quality standards specified in Section 3.4 of the Baseline Type A System Specification. The Testing Support Subsystem provides automated tools needed to minimize testing costs and to demonstrate conformance to the quality assurance standards specified in Section 4 of the Baseline Type A System Specification. The Configuration Management Support Subsystem provides automated tools needed to manage the System software and data configuration and to install the System on Government-specified host computers, as specified in Section 3.4 of the Baseline Type A System Specification. The Documentation Support Subsystem provides automated tools needed to aid in producing the quality software documentation required in Section 3.4 of the Baseline Type A System Specification.

To minimize potential development costs associated with the automated tools to be provided in the Support Complex, maximal use is made of automated tools provided by host computer operating systems. Although this approach naturally results in some Support Complex software differences between the two host computer families specified as a baseline for the Predesign Phase, resulting development and maintenance costs are minimized. These differences impact the architecture of the Support Complex at a very detailed level and therefore have no effect on the architecture presented in Figure 22 or in any other figure in Section 2.4.

#### **2.4.1 Development Support Subsystem**

The Development Support Subsystem aids the software development activities involved in generating the System software. These activities include not only gener-





**Figure 22. Hierarchical Definition of the Support Complex**

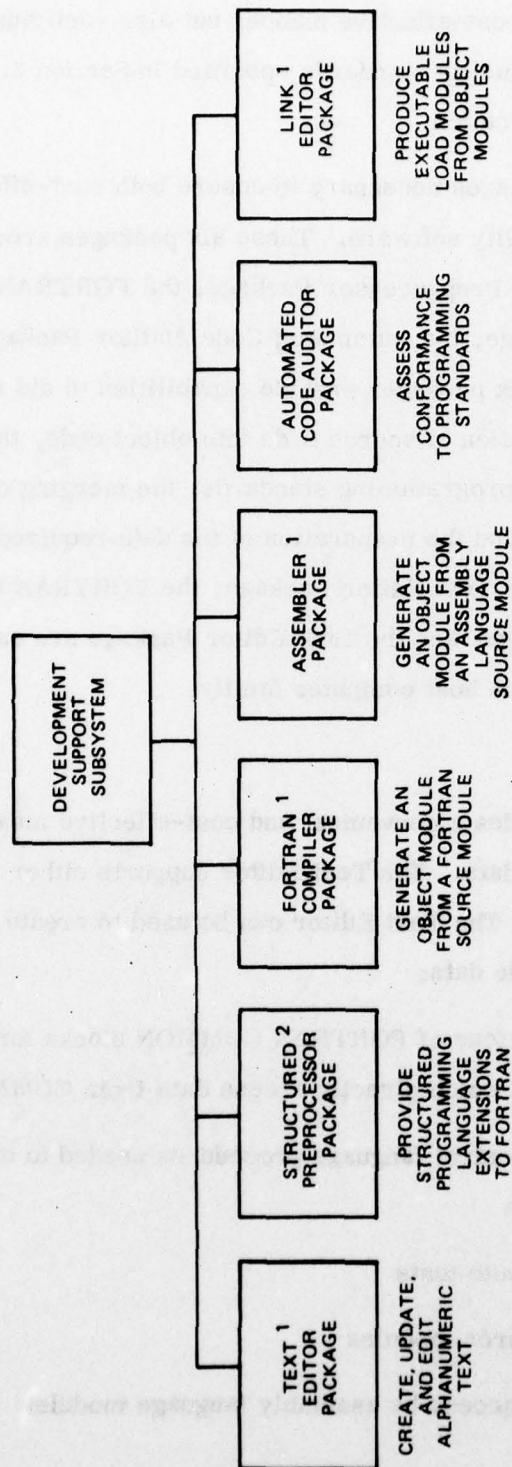
ating the System software in a cost-effective manner but also verifying that the software conforms to the high quality standards specified in Section 3.4 of the Baseline Type A System Specification.

Figure 23 presents the six packages necessary to ensure both cost-effective software development and high-quality software. These six packages are: the Text Editor Package, the Structured Preprocessor Package, the FORTRAN Compiler Package, the Assembler Package, the Automated Code Auditor Package, and the Link Editor Package. These six packages provide capabilities to aid in the generation of source code, the translation of source code into object code, the verification of source code conformance to programming standards, the merging of object code into executable load-modules, and the preparation of the data required by each of these support capabilities. The Text Editor Package, the FORTRAN Compiler Package, the Assembler Package, and the Link Editor Package are each composed of two subpackages, one for each host computer family.

#### **2.4.1.1 Text Editor Package**

The Text Editor Package provides a convenient and cost-effective means to create, update, and edit alphanumeric data. The Text Editor supports either an interactive or a batch user interface. The Text Editor can be used to create, update, and edit seven types of alphanumeric data:

1. Standardized definitions of FORTRAN COMMON blocks for use in FORTRAN modules which directly access data from COMMON blocks
2. Operating system control language procedures needed to execute elements of the System
3. Data needed to execute tests
4. Source code for source modules
5. Global macros for access by assembly language modules



<sup>1</sup>PART OF MODERN COMPUTER OPERATING SYSTEMS.

<sup>2</sup>AVAILABLE FROM COMMERCIAL SOFTWARE VENDORS.

Figure 23. Hierarchical Definition of the Development Support Subsystem



6. Link editor control statements needed to create System load-modules
7. Other user-defined alphanumeric text data (e.g., documentation)

The Text Editor Package is composed of one Text Editor Subpackage for each host development computer family. On a single host development computer, CSC recommends the use of a single Text Editor Subpackage: the one supplied by the host operating system.

#### 2.4.1.2 Structured Preprocessor Package

The Structured Preprocessor Package facilitates the use of top-down structured-programming concepts in FORTRAN source modules. This package provides ANSI-1966 FORTRAN language extensions which correspond to the following structured-programming control statements:

1. IF-THEN-ELSE
2. DO WHILE
3. DO UNTIL
4. DO FOR
5. CASE

The Structured Preprocessor Package translates structured-programming control statements to transportable FORTRAN statements and merges them with the remaining FORTRAN statements. The resulting FORTRAN module can then be compiled by the FORTRAN compiler. Thus FORTRAN modules can be written using structured-programming control statements despite the limitations of the FORTRAN compiler.

#### 2.4.1.3 FORTRAN Compiler Package

The FORTRAN Compiler Package translates ANSI-1966 FORTRAN source modules into object modules needed to build executable load modules. The FORTRAN Compiler Package is composed of one FORTRAN Compiler Subpackage for each host

computer family. On a single host computer, there will be a single FORTRAN Compiler Subpackage. This compiler will be provided by the host operating system.

#### 2.4.1.4 Assembler Package

The Assembler Package translates assembly language source modules into object modules needed to build load-modules. The Assembler Package allows the definition and utilization of both global and local macros. This package also permits access to FORTRAN named COMMON blocks.

The Assembler Package is composed of one Assembler Subpackage for each host computer family. On a single host computer, there is a single Assembler Subpackage. This subpackage is the Assembler provided by the host operating system.

#### 2.4.1.5 Automated Code Auditor Package

The Automated Code Auditor Package checks individual source modules for conformance to programming standards specified in Programming Standards for the Second-Generation Comprehensive Helicopter Analysis System.<sup>35</sup> For FORTRAN source modules, conformance is checked only for those standards which can be verified in a single scan of the source code. For assembly language source modules, only the format of the module prologue and the organization of module code are checked. Conformance to all remaining standards is verified via a visual inspection of the compilation or assembly listing. The size, complexity, transportability, and multicontractor development of the System require that programming standards be rigidly enforced. The use of an automated capability to verify

---

<sup>35</sup> PROGRAMMING STANDARDS FOR THE SECOND-GENERATION COMPREHENSIVE HELICOPTER ANALYSIS SYSTEM, Applied Technology Laboratory, U.S. Army Research and Technology Laboratories, Fort Eustis, Virginia, to be published.

conformance to specified programming standards minimizes the manual effort and, hence, the cost which would be needed if such an automated capability were not available.

#### 2.4.1.6 Link Editor Package

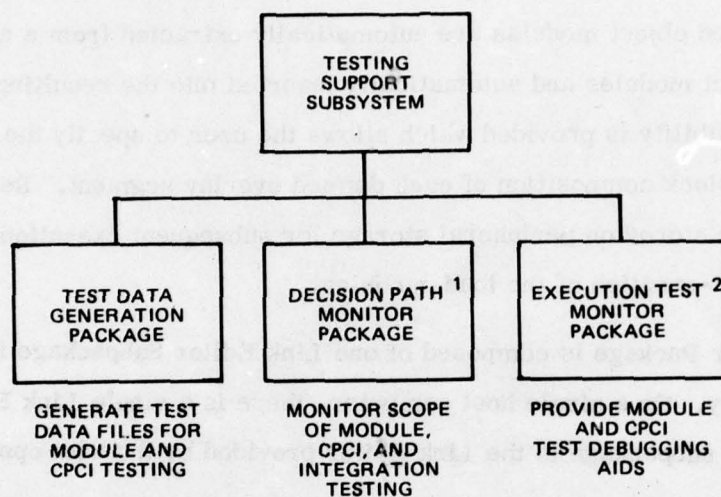
The Link Editor Package combines user-supplied object modules into executable load-modules. Object modules which are not user-supplied but which are invoked by user-supplied object modules are automatically extracted from a user-specified library of object modules and automatically inserted into the resulting load-module. An overlay capability is provided which allows the user to specify the module and the COMMON block composition of each defined overlay segment. Resulting load-modules can be stored on peripheral storage for subsequent execution without necessitating a re-creation of the load-modules.

The Link Editor Package is composed of one Link Editor Subpackage for each host computer family. On a single host computer, there is a single Link Editor Subpackage. This subpackage is the Link Editor provided by the host operating system.

#### 2.4.2 Testing Support Subsystem

The Testing Support Subsystem provides automated tools for module, CPCI, and integration testing, both to reduce testing costs and to demonstrate conformance to the quality assurance provisions specified in Section 4 of the Baseline Type A System Specification. Figure 24 presents the three testing tools provided by the Testing Support Subsystem. These three tools correspond to three software packages: the Test Data Generation Package, the Decision Path Monitor Package, and the Execution Test Monitor Package. The Test Data Generation Package and the Execution Test Monitor Package both contribute toward reducing the costs of testing by simplifying the tasks of generating test data and of debugging software code. The Decision Path Monitor Package provides an objective assessment of the scope of module, CPCI, and integration testing.





<sup>1</sup> AVAILABLE FROM COMMERCIAL SOFTWARE VENDORS.

<sup>2</sup> PART OF MODERN COMPUTER OPERATING SYSTEMS.

**Figure 24. Hierarchical Definition of the Testing Support Subsystem**

#### 2.4.2.1 Test Data Generation Package

The Test Data Generation Package simplifies the task of providing test data for use in testing modules or CPCIs. This package provides a quick and convenient means of formatting and organizing test data for direct use by a module or a CPI. The generated test data is derived from user-supplied (i.e., flight test) data, from data extracted from either the Master Data Base or the Master Command File, or from an already existing file of test data. The generated test data is formatted and organized so that it can be accessed at test-execution time by using the software services provided by the Data Base Management Subsystem of the Operational Complex. This frees the package user from any dependency on the format and organization of the generated file of test data, thus minimizing the need to generate special software to test a module or a CPI.

#### 2.4.2.2 Decision Path Monitor Package

The Decision Path Monitor Package monitors the execution of decision paths and module invocations during tests. This capability is used to objectively determine the scope of testing performed. Decision path monitoring is used to determine the scope of module tests; module invocation monitoring is used to determine the scope of CPI and integration tests.

The Decision Path Monitor Package identifies decision paths and module invocations in FORTRAN source modules; inserts invocations (CALLs) to instrumentation data collection modules at the beginning of each identified decision path and immediately preceding each module invocation in the original source module; and identifies, based on the collected instrumentation data, the number of times each decision path and module invocation was executed during a test. A decision path is defined for each serial sequence of instructions which can be executed following a conditional branch or loop definition statement. Module invocations include both direct module invocations (e.g., FORTRAN CALL statement) and indirect module invocations (e.g., FORTRAN external function reference).

The Decision Path Monitor Package provides an objective means to determine whether the scope of module, CPCI, and integration tests conforms to the standards specified in Section 4 of the Baseline Type A System Specification. It is used both to determine whether all decision paths have been tested in each module and to determine whether all module invocations have been executed in a CPCI (CPCI testing) and in a collection of CPCIs (integration testing). Without the objective, quantitative monitoring capability of the Decision Path Monitor Package, the assessment of the scope of module, CPCI, and integration testing would have to rely on qualitative and often imprecise human judgments.

#### **2.4.2.3 Execution Test Monitor Package**

The Execution Test Monitor Package provides interactive and batch debugging capabilities for testing individual modules and CPCIs of the System. The Execution Test Monitor Package provides specific capabilities to request

1. Intermediate execution data displays for user-specified variables at user-selected points within modules
2. Intermediate dumps of selected portions of memory
3. Trace of all module invocations and contents of associated calling sequence parameters

The Execution Test Monitor Package uses execution-time debugging capabilities provided by the host computer operating system. This reliance on host operating system debugging capabilities results in multiple Execution Test Monitor Subpackages, one for each host computer family. On a single host computer, there is only one Execution Test Monitor Subpackage: the one provided by the host operating system.

#### **2.4.3 Configuration Management Support Subsystem**

The Configuration Management Support Subsystem provides automated support both for controlling the System software and data configuration and for installing the

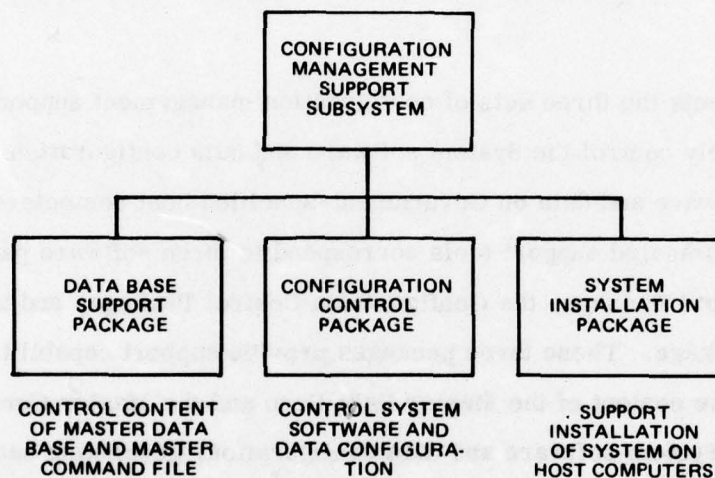


System software and data on Government-specified host computers, as specified in Section 3.4 of the Baseline Type A System Specification. For software systems, configuration control must address documentation (development and product specifications), software, and data. Although controlling the documentation elements of System configuration lends itself to manual procedures, such is not the case with System software and data. Successful management of System software and data configuration requires that manual procedures be supported by automated tools. It is this requirement that is supported by the Configuration Management Support Subsystem.

Figure 25 presents the three sets of configuration management support tools needed both to effectively control the System software and data configuration and to install the System software and data on Government-specified host computers. These three sets of automated support tools correspond to three software packages: the Data Base Support Package, the Configuration Control Package, and the System Installation Package. These three packages provide support capabilities which aid in controlling the content of the Master Data Base and the Master Command File; controlling the System software and data configuration; generating backup copies of all System software libraries and data files; restoring System software libraries and data files; generating installation tapes for System software libraries and data files; and installing the System on another host computer.

#### 2.4.3.1 Data Base Support Package

The Data Base Support Package controls the content of the subset of System data reflected by the Master Data Base and the Master Command File. Because the content of the Master Data Base and the Master Command File are critical to an effective, orderly utilization of the Operational Complex, controls are provided over all modifications to them. This package maintains audit trails for all modifications. Special authorization codes, which are under control of each installation, are required before modifications are permitted. The control over the content of the Master Data Base and the Master Command File at a computer



**Figure 25. Hierarchical Definition of the Configuration Management Support Subsystem**

facility is intended to be vested in a single group of data base administrators. All modifications to the Master Data Base and the Master Command File are made by the data base administrators. (Although CSC strongly recommends this type of control, there are no explicit or implicit software design assumptions that require a user organization to exercise it.) It is because of this centralization that the Data Base Support Package provides rigid controls over all modifications to the Master Data Base and the Master Command File. The Data Base Support Package also provides a capability to generate reports showing the content of the Master Data Base and Master Command File.

#### 2.4.3.2 Configuration Control Package

The Configuration Control Package provides the following tools needed to control the System software and data configuration, in accordance with the System Development Plan (see Section 5):

1. Maintenance of multiple versions of the System
2. Protection from unauthorized modification of all System software libraries and data files
3. Generation of backup copies of System software libraries and data files
4. Maintenance of both a record and a copy of all authorized System software and data modifications
5. Restoration of System software libraries and data files from backup copies

The Configuration Control Package uses, to the maximum possible extent, operating system utilities which are available on each Government-specified host computer family to provide configuration control tools. This reliance on host computer operating system utilities results in one Configuration Control Subpackage for each host computer family. On a single host computer, there is a single Configuration



Control Subpackage based upon configuration control tools provided by the resident host operating system.

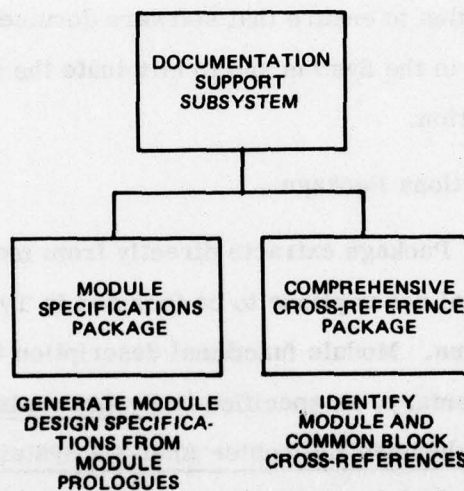
#### 2.4.3.3 System Installation Package

The System Installation Package provides automated tools to support the installation of the System on Government-specified target computers. All System software and data needed for installation is generated in magnetic tape format on a development computer. The resulting magnetic tapes are then used on the target computer to create the needed System software libraries and data files.

The System Installation Package uses, to the maximum possible extent, operating system utilities which are available on both the development computer and the target computer (i.e., the computer on which the System is installed). This reliance on host computer operating system utilities results in one System Installation Subpackage for each target host computer family. Each subpackage supports the installation of the System from one host computer to another host computer in the same family. In addition, the capability is provided to install, on a host computer which is not a member of the primary (Host 1) computer family, the transportable System software and data developed on the primary (Host 1) development computer. This additional capability is provided as part of the System Installation Subpackage for each target host computer family which differs from the primary (Host 1) development computer family.

#### 2.4.4 Documentation Support Subsystem

The Documentation Support Subsystem provides automated support tools for generating software documentation during System development and maintenance. Figure 26 presents the two basic sets of software documentation support tools provided by this subsystem: the Module Specifications Package and the Comprehensive Cross-Reference Package. Both the Module Specifications Package and the Comprehensive Cross-Reference Package are transportable; hence they do not have



**Figure 26. Hierarchical Definition of the Documentation Support Subsystem**

host computer-dependent subpackages. The Module Specifications Package extracts module design specifications directly from prologue commentary included in module source code for direct use in Type C5 Computer Program Product Specifications. The Comprehensive Cross-Reference Package extracts the module and COMMON block cross-reference data direct from module source code for inclusion in the dictionary of computer variables specified in Section 3.4.1.5 of the Baseline Type A System Specification. The intent of the System Documentation Subsystem is to generate as much of the software documentation as possible directly from source modules to ensure that software documentation reflects the software actually included in the System and to eliminate the potential error inherent in human transcription.

#### 2.4.4.1 Module Specifications Package

The Module Specifications Package extracts directly from module source code the individual module functional descriptions to be included in Type C5 Computer Program Product Specifications. Module functional description information is included in module prologue commentary, as specified in Programming Standards for the Second-Generation Comprehensive Helicopter Analysis System.<sup>35</sup> For each module, a prologue includes a module description, a logic flow summary via structured design constructs in combination with English language phraseology, data and module interface specifications, data organization diagrams, and processing limitations. Because this information includes all that is required for module functional descriptions in a Type C5 Computer Program Product Specification, a separate and independent documentation effort for System modules is avoided by using the Module Specifications Package. This ensures that the published module functional specifications remain up to date with the actual source code. In addition, the use of the Module Specifications Package avoids the very costly effort of generating and maintaining separate module functional description documentation.



#### 2.4.4.2 Comprehensive Cross-Reference Package

An important element of documentation specified in the Baseline Type A System Specification is a dictionary of computer variables. Included in this dictionary is an entry for every module and COMMON block. For each module and COMMON block entry, a list of all modules which reference the entry is required. The Comprehensive Cross-Reference Package directly extracts this cross-reference information from module source code. This package eliminates the manual effort necessary to derive this information and automatically ensures that the information is complete, correct, and up to date.

The cross-reference information is derived from specification statements and executable statements in FORTRAN source modules and from the information included in the prologue of assembly language source modules. In FORTRAN source modules, module invocations include both modules directly invoked via a CALL statement and modules invoked via an external function reference. Two types of cross-reference information are output by this package: (1) a report by module which indicates by source statement all references to external modules and all definitions of COMMON blocks and (2) a report by external reference (module or COMMON block) which identifies all modules which reference the module or COMMON block.

#### 2.5 SYSTEM COMMAND SEQUENCES AND DATA FLOW

This section discusses the sequence of functions to be performed and the flow of data necessary for the solution of several typical engineering problems. As indicated in Section 2.3, Executive Component software progresses through entries in the Sequence Control Table. This is an internal System table constructed during the input phase of an analysis run by the User Input Package of the Executive Component from two sources: user input and the Master Command File. These Sequence Control Table entries indicate which Technology Component or Executive Component software elements (subpackages or packages) are to be executed and in

what sequence. Information in the Sequence Control Table also provides the looping and branching logic required for the proper execution of the software elements.

The Sequence Control Table for a given engineering case indicates a specific execution sequence. This execution sequence, called a System Command Sequence, is made up of two elements: subsequences and logic controlling the execution of the subsequences.

A subsequence is a set of System Commands which performs a single, major System function (e.g., finding a steady-state flight condition, calculating stability and control data, calculating a maneuver, calculating loads and vibrations). Subsequences serve two main purposes. First, they facilitate understanding of System operation for all involved--the Government, the developer, and users. This leads to better communication, fewer errors, and reduced costs. Secondly, their use reduces the difficulty of generating the System Command Sequences. In fact, it enables the User Input Package to interpret a rather small set of user input keywords and construct a valid Sequence Control Table for the physical configuration desired. This flexibility is obtained without placing a great burden upon the user.

Subsequences may be thought of as building blocks that may be combined in a variety of ways to create System Command Sequences. Subsequences have been grouped together into sets (types) to facilitate understanding the relationships between subsequences and to facilitate the utilization of subsequences in building System Command Sequences. Eight sets of subsequences have been identified. Subsequences within a set differ from each other only in terms of the procedure employed to perform the function and/or the level of detail at which the function is to be performed. Table 1 identifies the eight sets of subsequences and the function of each. For example, three Trim Subsequences are currently identified, each of which is used to solve for a steady-state flight condition. The major difference

Table 1. Subsequence Sets and Functions

SUBSEQUENCE SETS	FUNCTION
INITIALIZATION	INITIALIZES COORDINATE SYSTEMS AND TRANSFORMATIONS AND CONSTANT VALUES IN THE EQUATIONS OF MOTION; CHECKS DATA COMPATIBILITY; ESTABLISHES COMPONENT INTERFACES; AND INITIALIZES DYNAMIC COUPLING
SIMULATION MODEL	CALCULATES ALL OF THE MATRIX ELEMENTS AND FORCING FUNCTIONS IN THE EQUATIONS OF MOTION AS A FUNCTION OF TIME OR AZIMUTH
TRIM	SOLVES FOR A STEADY-STATE FLIGHT CONDITION USING USER-SPECIFIED DEGREES OF FREEDOM AND CONSTRAINTS
MANEUVER	CALCULATES A TIME HISTORY WITH USER-SPECIFIED CONTROL MOTIONS, GUSTS, PRESCRIBED RESPONSE, AND FAILURE/DAMAGE EFFECTS
STABILITY AND CONTROL	CALCULATES THE STABILITY AND CONTROL CHARACTERISTICS FOR AN AIRCRAFT CONFIGURATION IN EITHER STEADY-STATE OR ACCELERATED FLIGHT
LOADS AND VIBRATIONS	CALCULATES THE SHEARS, MOMENTS, STRESSES, DISPLACEMENTS, VELOCITIES, AND ACCELERATIONS AT USER-SELECTED NODE POINTS ON THE PHYSICAL CONFIGURATION
ACOUSTICS	CALCULATES THE INTERNAL NEAR-FIELD AND FAR-FIELD SOUND LEVELS AS A FUNCTION OF FLIGHT CONDITION AND GROUND SURFACE
AEROELASTIC STABILITY	SOLVES FOR THE AEROELASTIC FREQUENCIES, DAMPING, AND MODE SHAPES AS A FUNCTION OF PHYSICAL CONFIGURATION AND FLIGHT CONDITION



among the three Trim Subsequences is the procedure employed to solve for a steady-state flight condition, i.e., the Simultaneous Iterate-To-Trim procedure, the Decoupled Fly-To-Trim procedure, or the Fly-To-Trim procedure.

#### 2.5.1 System Command Sequences

By using subsequences, all of the problems which must be solved by the System may be encompassed by only five System Command Sequences, one for each of the five aircraft technical characteristics (i.e., performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability). These five sequences are discussed in detail, including the subsequences used and the flow of data, in Sections 2.5.1.1 through 2.5.1.5.

##### 2.5.1.1 Performance System Command Sequence

The Performance System Command Sequence, shown in Figure 27, is the simplest of the five System Command Sequences. The center portion of the figure illustrates how the execution processes proceed from one step to another. (Each process block has a number associated with it for discussion purposes.) The flow of processing control is shown by solid arrows. The left portion of the figure lists sets of data which are input to the various processing blocks, as indicated by dashed arrows. These data sets are in the same form as the user's input (except for change of units or straightforward algebraic combinations. For some subsequences these sets will have been produced by other subsequences. The right portion of the figure lists data sets which are output by the process blocks. Data sets output by one process may be input to a later process; dashed lines show this. The data sets in both the input and output columns of this and similar diagrams that follow are the same as those defined in Appendix A of the Predesign Phase Type B5 Development Specifications for the Second-Generation Comprehensive Helicopter Analysis System. User output (e.g., intermediate and final printed or plotted output) is controlled by the Executive Component, depending

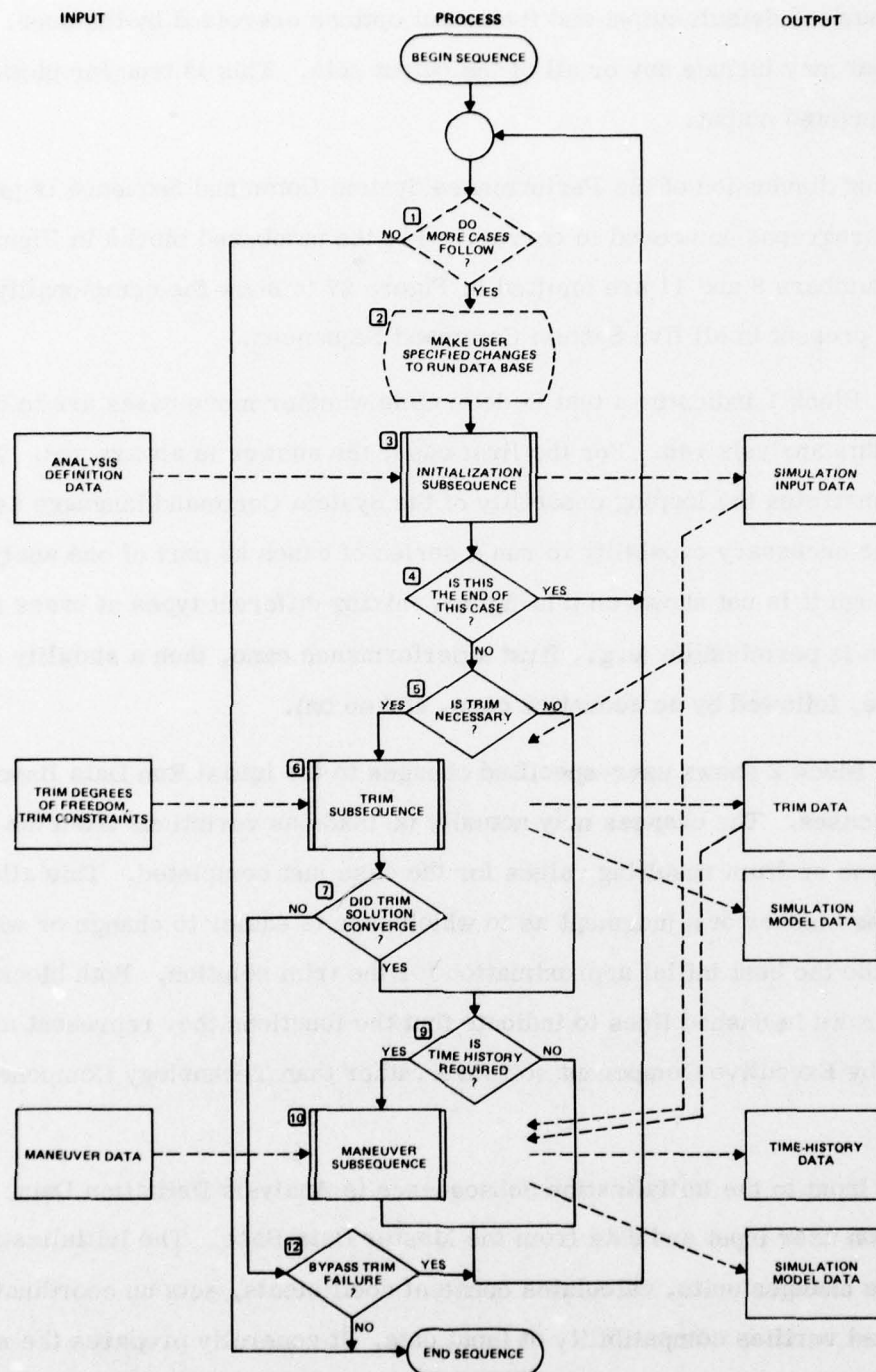


Figure 27. Performance System Command Sequence

upon the standard default output and the output options exercised by the user. The user's output may include any or all of the output sets. This is true for plotted as well as printed output.

The following discussion of the Performance System Command Sequence is presented in paragraphs numbered to correspond to the numbered blocks in Figure 27. (The numbers 8 and 11 are omitted in Figure 27 to show the commonality of processing present in all five System Command Sequences.)

1. Block 1 indicates a test to determine whether more cases are to be run as part of this analysis run. For the first case, the answer is always yes. This block demonstrates the looping capability of the System Command language as it provides the necessary capability to run a series of cases as part of one analysis run. Although it is not shown on this figure, mixing different types of cases in one analysis run is permissible (e.g., first a performance case, then a stability and control case, followed by an acoustics case, and so on).

2. Block 2 shows user-specified changes to the initial Run Data Base for succeeding cases. The changes may actually be made as variations from the initial input case or from resulting values for the case just completed. This allows a user to use his/her own judgment as to which case is easier to change or which would provide the best initial approximation for the trim solution. Both blocks 1 and 2 are drawn in dashed lines to indicate that the functions they represent are performed by Executive Component software rather than Technology Component software.

3. Input to the Initialization Subsequence is Analysis Definition Data, which is based upon user input and data from the Master Data Base. The Initialization Subsequence changes units, calculates constant coefficients, sets up coordinate systems, and verifies compatibility of input data. It generally prepares the major set of data known as Simulation Input Data. Rotor modes and initial geometric



shape of the rotor wake are calculated in this subsequence when these functions are needed. All of the data compatibility checks are completed in this subsequence.

4. Block 4 indicates a test to determine whether the case ends at this point. If the purpose of this case was to check the input data or to find rotor natural frequencies and mode shapes, then the case is complete, and control returns to block 1.

5. Block 5 indicates a test to determine whether a trim condition must be found for this case. If so, as is normally the case, control passes to block 6. However, several conditions exist for which a trim is not required. If the data changes made between this case and a previous case do not affect the trim condition, using the previously calculated trim condition is more efficient. An example of such a data change is a gain or time constant in an automatic flight control system.

6. The Trim Subsequence, which solves for a steady-state flight condition according to the user-specified trim degrees of freedom and trim constraints, makes use of (i.e., invokes) the Simulation Model Subsequence in its solution. The output sets are the Trim Data and the Simulation Model Data for the trim condition. These sets may be saved in a file for subsequent processing or plotting, at the user's option.

7. Block 7 indicates a test on the convergence of the trim. If the trim solution attempt has taken too many iterations or has taken too much time, or if some other convergence problem was discovered, this case ends and control passes to block 12. For a successful trim, control is transferred to block 9.

9. Block 9 indicates a test to determine whether a time-history solution is required. This test, based on the user's input for maneuvers, may indicate that it is unnecessary even to invoke the Maneuver Subsequence.

10. The Maneuver Subsequence generates a time-history solution of the differential equations of motion. This process is controlled by one of the subpackages of the Differential Equations Solution Package of the General Mathematical Operations Subsystem. The user's input control motions, gust disturbances, and other such data are implemented by packages of the Maneuver Subsystem. All or part of the Simulation Model Data generated may be saved in the Time-History Data set for later processing or plotting. From block 10 control passes back to the top of the sequence, block 1, to determine whether more cases follow.

12. Block 12 is a test which occurs following a trim that did not converge. Normally, if one trim fails, the following cases fail also, and the analysis run thus terminates. However, in special cases, each trim is somewhat independent of the others in the run. In these cases the user has available an override control input which returns control to the top of the sequence to execute any cases which follow.

#### 2.5.1.2 Stability and Control System Command Sequence

The Stability and Control System Command Sequence is shown in Figure 28. The following discussion of this sequence is presented in the format used in Section 2.5.1.1. Discussion of blocks 1 through 7, 9, and 12 in Figure 28 is omitted because these blocks are identical to blocks 1 through 7, 9, and 12 in Figure 27.

8. The Stability and Control Subsequence consists of System Commands which invoke packages of the Stability and Control Subsystem to calculate the aircraft flight stability characteristics, including complex eigenvalues and eigenvectors for evaluating controls-fixed stability, and transfer functions and frequency response for step control input and harmonic control input (subsets of the Stability and Control Data set). This solution is based on linear perturbation equations obtained by use of the Simulation Model Subsequence.

10. At user-specified times during the time history, data are saved for subsequent use by the Stability and Control Subsequence.

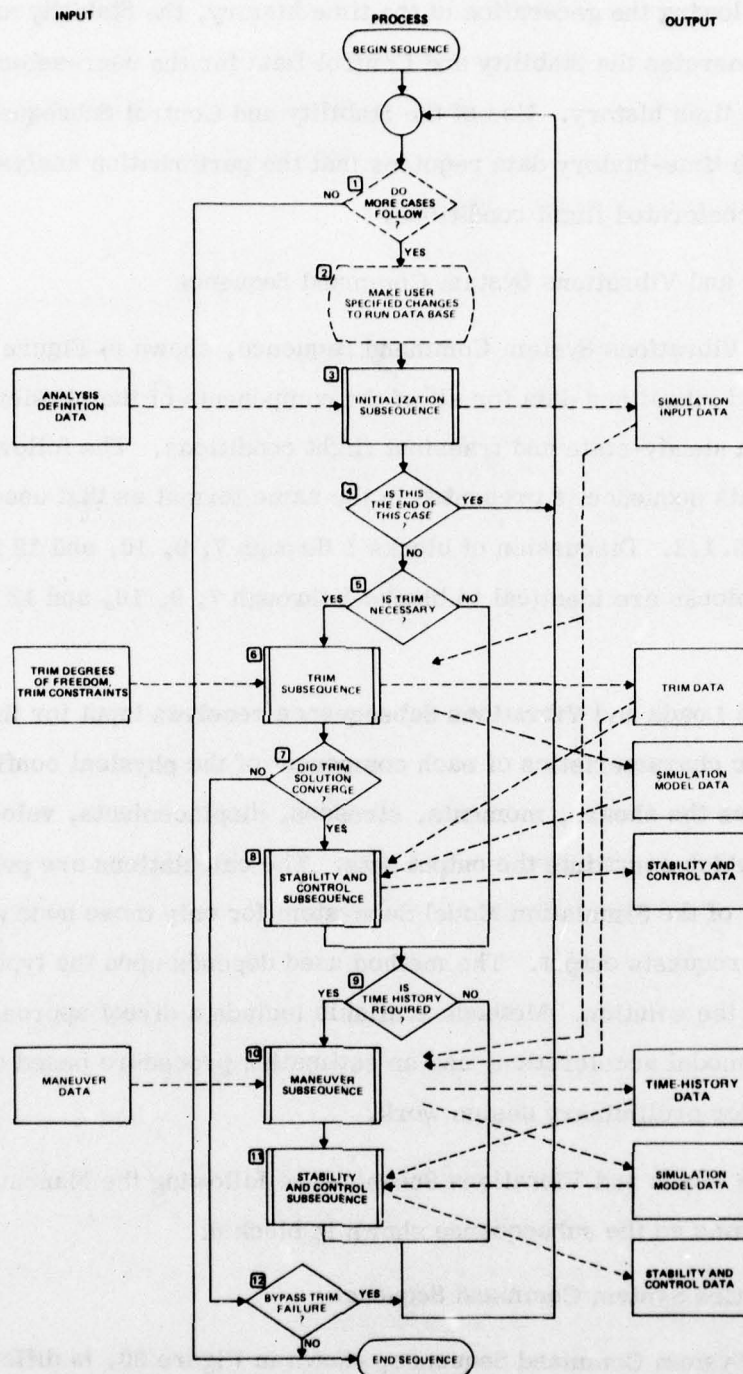


Figure 28. Stability and Control System Command Sequence



11. Following the generation of the time history, the Stability and Control Subsequence generates the Stability and Control Data for the user-selected time points from the time history. Use of the Stability and Control Subsequence in conjunction with time-history data requires that the perturbation analysis must be applicable to accelerated flight conditions.

#### 2.5.1.3 Loads and Vibrations System Command Sequence

The Loads and Vibrations System Command Sequence, shown in Figure 29, generates loads and vibrations data for all of the components of the physical configuration for both steady-state and transient flight conditions. The following discussion of this sequence is presented in the same format as that used in Sections 2.5.1.1 and 2.5.1.2. Discussion of blocks 1 through 7, 9, 10, and 12 is omitted because these blocks are identical to blocks 1 through 7, 9, 10, and 12 in Figure 28.

8. The Loads and Vibrations Subsequence receives input for the response and the dynamic characteristics of each component of the physical configuration. It then calculates the shears, moments, stresses, displacements, velocities, and accelerations which constitute the output sets. The calculations are performed by subpackages of the Simulation Model Subsystem for only those node points at which the user requests output. The method used depends upon the type of dynamic model used for the solution. Methods available include a direct approach, modal displacement, modal acceleration, and an estimating procedure based on amplification factors for preliminary design work.

11. The Loads and Vibrations Subsequence following the Maneuver Subsequence is the same as the subsequence shown in block 8.

#### 2.5.1.4 Acoustics System Command Sequence

The Acoustics System Command Sequence, shown in Figure 30, is different from the three previously discussed sequences only with respect to process blocks 8

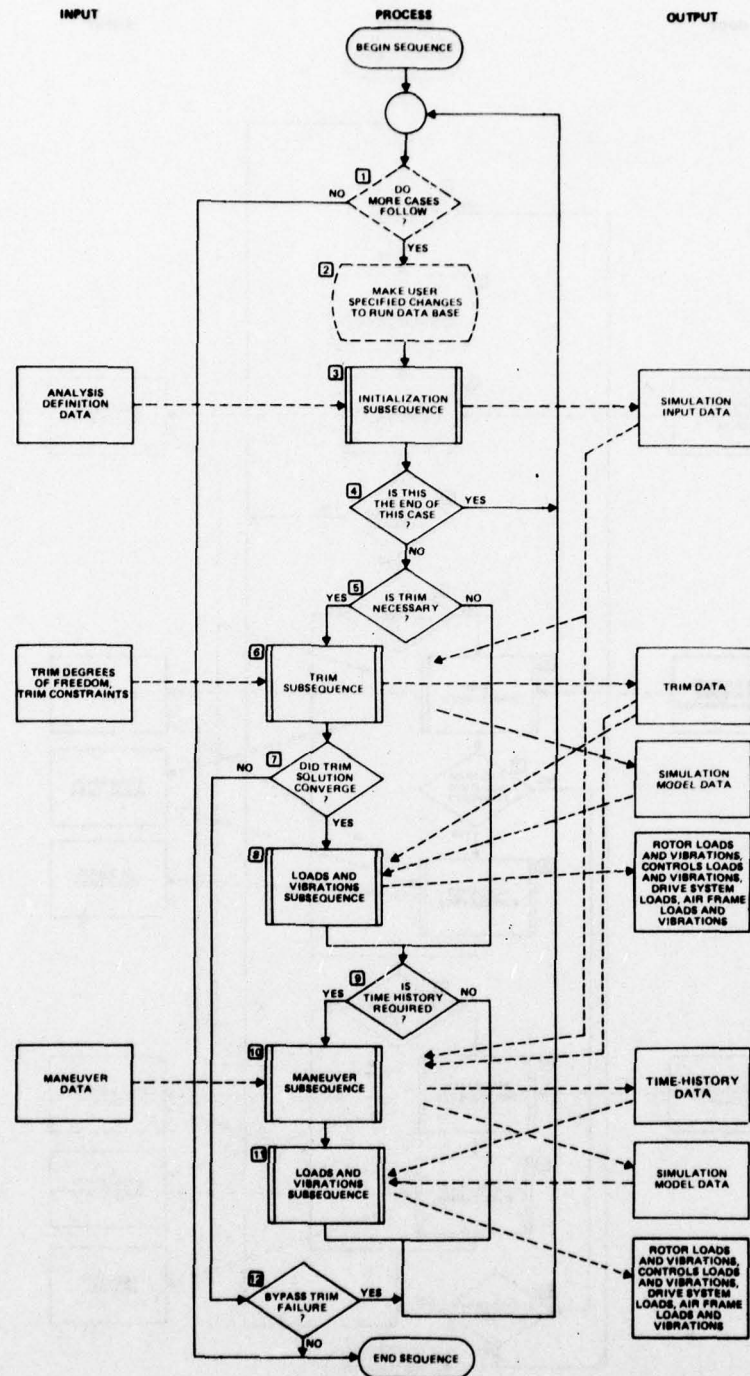


Figure 29. Loads and Vibrations System Command Sequence

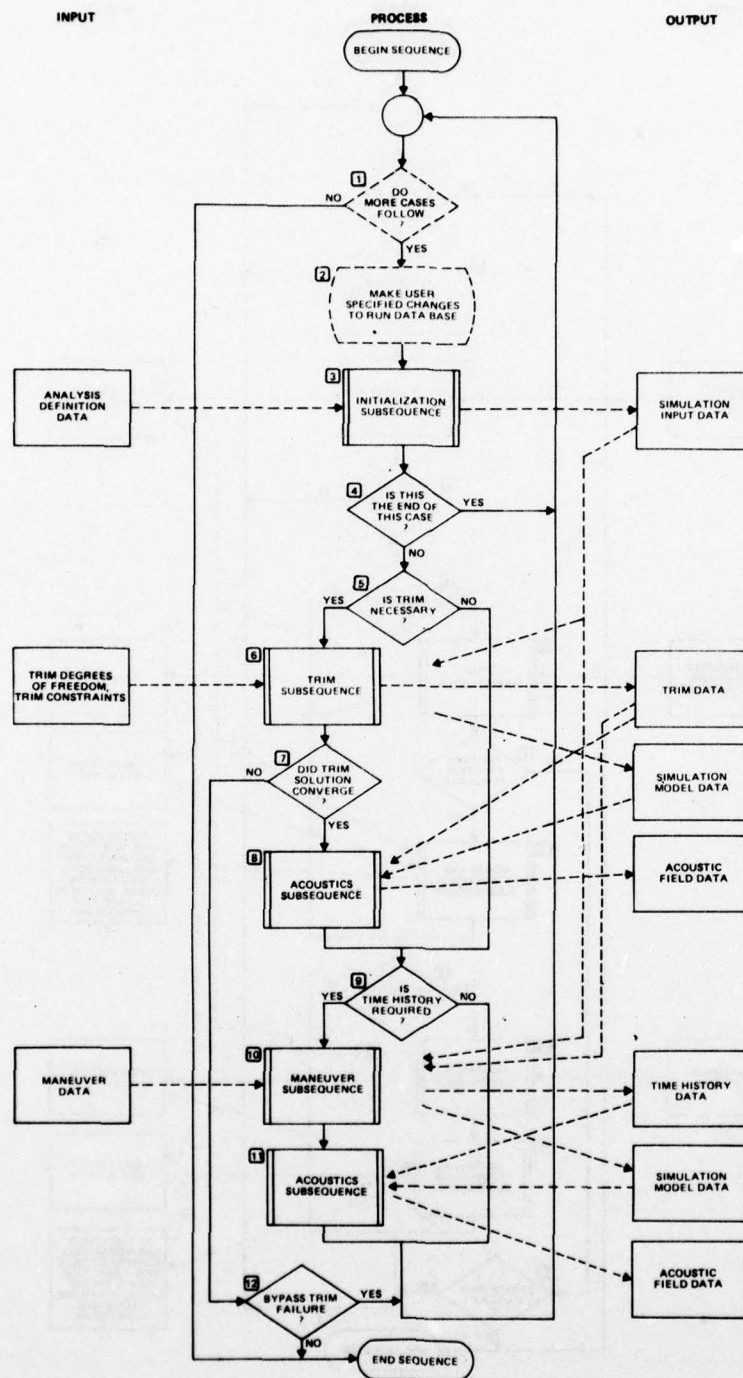


Figure 30. Acoustics System Command Sequence



and 11, which contain the Acoustics Subsequence. The Acoustics Subsequence makes use of the packages of the Acoustics Subsystem to calculate the internal and near-field and far-field sounds created by the aircraft configuration. These data appear in the Acoustic Field Data set. This subsequence represents several levels of complexity consistent with the levels of complexity used in the Simulation Model Subsequence. Methods vary from purely empirical methods to considerations of vortex shedding and vortex interference effects.

#### 2.5.1.5 Aeroelastic Stability System Command Sequence

The Aeroelastic Stability System Command Sequence, shown in Figure 31, is different from the four previously discussed sequences only in regard to blocks 8 and 11. These blocks show the Aeroelastic Stability Subsequence, which calculates the Aeroelastic Stability Data. This set of data includes aeroelastic frequencies, damping, and mode shapes. This process may be performed by any of three optional methods that make use of one of the three packages of the Aeroelastic Stability Subsystem (i.e., the Linear Aeroelastic Stability Analysis Package, the Floquet Analysis Package, or the Aeroelastic Stability Postprocessing Package).

Because of the similarity in structure of the five System Command Sequences, the combination of more than one technical characteristic in a single case is possible. For example, referring to Figure 31, the Loads and Vibrations Subsequence may be inserted before block 8, the Aeroelastic Stability Subsequence, if the user wishes to obtain loads and vibrations data as well as aeroelastic stability data for a particular case.

#### 2.5.2 Details of Subsequences

The use of subsequences has been shown to be an easily understood yet powerful tool for solving engineering problems. For further understanding it is necessary to see how subsequences are constructed and how they function. One Trim Subsequence and two Simulation Model Subsequences are presented in the following subsections to convey this understanding. The Trim Subsequence demonstrates

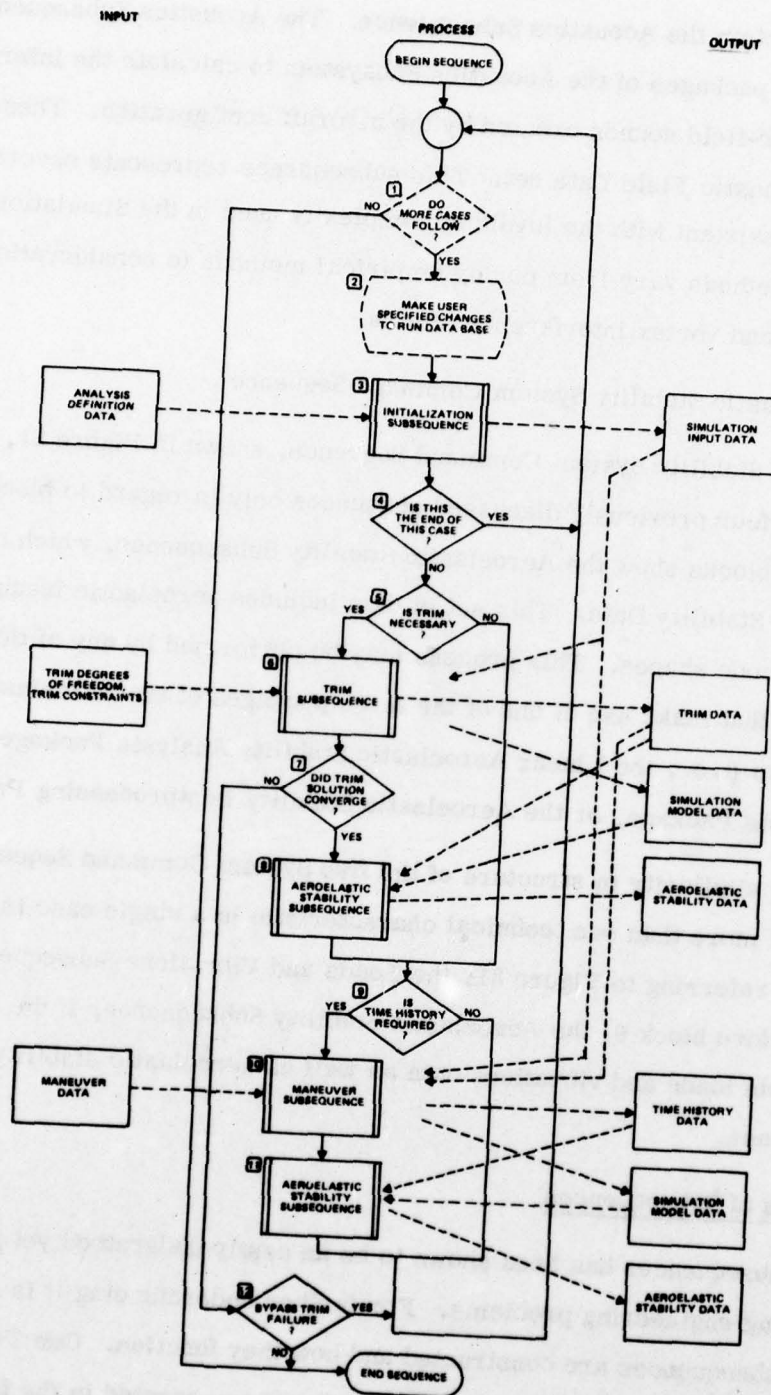


Figure 31. Aeroelastic Stability System Command Sequence

how a complex logic network functions within a System. The two Simulation Model Subsequences show how the various levels of complexity enter the analysis.

#### 2.5.2.1 Trim Subsequence

The Trim Subsequence shown in Figure 32 shows the flow of processing and data for the Simultaneous Iterate-to-Trim Package of the Trim Subsystem. The processing blocks shown are as follows:

1. The Set Up Trim process employs the user-specified Trim Degrees of Freedom and Trim Constraints sets to form the steady-state equations which must be solved. This includes fixed variables, independent variables, and the form and number of the equilibrium equations to be satisfied. The primary result is the Old Approximate Trim Solution, from which an improved solution is to be generated.
2. This block indicates the top of a loop which calculates the Simulation Model Data for several azimuth locations which represent one rotor revolution. When all azimuth locations have been calculated, control is passed to block 4 to test for trim convergence.
3. The Simulation Model Subsequence calculates all of the matrix coefficients and forcing functions for the equations of motion of the configuration. The details of this subsequence are discussed in Sections 2.5.2.2 and 2.5.2.3 for two different levels of complexity: preliminary design and detailed design.
4. Block 4 indicates a test of the Trim Convergence Criteria. This may be a rather complicated set of tests, depending upon the complexity of the aircraft configuration. Therefore, this series of tests is designed as a subpackage which sets a single parameter to be interrogated in the System Command Sequence.
5. The process of block 5 is contained in the same subpackage as block 4. When trim convergence is achieved, the Trim Convergence Indicator is set to TRUE, and the trim solution values are made available for further processing.



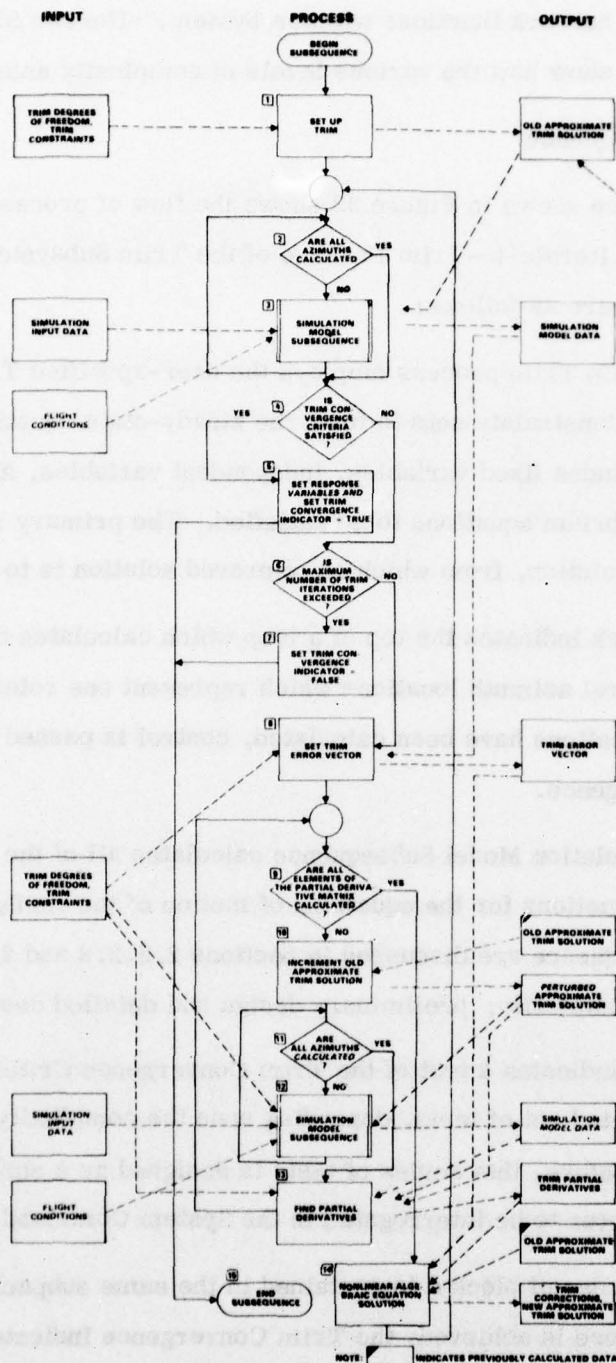


Figure 32. Simultaneous Iterate-To-Trim Subsequence

6. If the trim solution has not converged at this iteration, the iteration counter is incremented, and tests are made to determine whether the maximum number of iterations has been exceeded.

7. If the maximum number of trim iterations has been exceeded, the Trim Convergence Indicator is set to FALSE. This indicates to the Run-Time Control Package that the trim solution has failed.

8. The Set Trim Error Vector Subpackage sets an array which is a measure, based on the Simulation Model Data calculated in block 3, of how far the Old Approximate Trim Solution is from the New Approximate Trim Solution.

9. Block 9 indicates a test to determine whether all of the elements of the Trim Partial Derivatives set have been calculated. This is the controlling test for a loop which calculates partial derivatives of forcing functions for each of the independent variables.

10. This block represents the Increment Old Approximate Trim Solution Subpackage. It makes prescribed increments to each of the independent degrees of freedom, one at a time, as the looping controlled by block 9 continues.

11, 12. These two blocks again indicate an azimuth loop to calculate the Simulation Model Data using both the Perturbed Approximate Trim Solution and the Simulation Input Data.

13. The Find Partial Derivatives block calculates the change in the variables of the Trim Error Vector from block 12, minus the values of the Trim Error Vector set in block 8, divided by the increment to the particular degree of freedom, to obtain the Trim Partial Derivatives set.

14. The Nonlinear Algebraic Equation Solution Package is a part of the General Mathematical Operations Subsystem. It is used to generate a set of Trim Corrections to the independent variables which leads to the New Approximate Trim Solution. Control then returns to the top of the subsequence to block 2, where the

New Approximate Trim Solution (rather than the Old Approximate Trim Solution) is used as input to the Simulation Model Subsequence.

#### 2.5.2.2 A Simulation Model Subsequence for Preliminary Design

A Simulation Model Subsequence generates the mass, damping, and stiffness matrices and the forcing functions representing the aircraft configuration and its environment. The environment includes analysis components other than the aircraft, such as airmass, wind tunnel, test stand, and ground/deck reference. A Simulation Model Subsequence comprises System Commands which invoke subpackages of the Simulation Model Subsystem. Because this subsequence is used in every System Command Sequence, its importance cannot be overemphasized. Therefore, a considerable amount of effort has been spent to provide the user with maximum flexibility to specify different types of models while at the same time requiring him to know very little about the internal workings of the System. This provision results in a minimum amount of time and effort in generating models, in specifying solution requests, and in obtaining answers. This is partially accomplished by specifying the levels of complexity and the solution methods in very brief English-language-type commands. In addition, many of the subpackages used in this subsequence fall into logical groups so that a single user input in the form of a meaningful keyword activates a logical group of subpackages which performs analysis for a single aircraft component. These logical groups may be thought of as sub-sequences.

The use of such groups eliminates many of the possible interface problems between subpackages. Considering the foregoing, only a limited number of commonly used versions of the Simulation Model Subsequence need be defined for user convenience. It is not necessary to have all combinations of Simulation Model Subsequences built into the Master Command File because the User Input Package has the capability to dynamically build the sequence based upon user input data. The subsequences defined for user convenience include one for preliminary design and



five each for detailed design and for research. These five include common aircraft configurations: (1) single-rotor helicopter, (2) tandem-rotor helicopter, (3) coaxial helicopter, (4) tilt-rotor aircraft, and (5) test stand/wind tunnel configuration.

The Simulation Model Subsequence for preliminary design is shown in Figure 33. The following discussion of this sequence is presented in paragraphs numbered to correspond to the numbered blocks or dotted lines in the figure. Each of the rectangular processing blocks represents one of the Executive-invokable subpackages of the Simulation Model Subsystem.

1. Dotted line 1 is a simple example of the selections made by the User Input Package of the Executive Component in building the Sequence Control Table. If the user chooses a single-rotor helicopter configuration, the Standard Rigid Control System Subpackage is included in the subsequence by the User Input Package to calculate the Standard Rigid Control System. For other user-specified configurations which require more complex control linkages, the Generalized Coupling Rigid Control System Subpackage is placed in the Sequence Control Table by the User Input Package.

2, 3. For the appropriate configuration, one of two subpackages, i.e., the Standard Rigid Control System Subpackage or the Generalized Coupling Rigid Control System Subpackage, computes the Control Settings set which contains the local control angles for all controllable components (rotors, aerodynamic surfaces, stores, or other).

4. The Rigid Constant Speed Drive Subpackage calculates the rotational speeds of all drive shafts by use of a reference rotational speed in the Flight Conditions set and the geometry concerned.

5. Block 5 calculates the Fluid Flow Field in terms of absolute velocity components at every aerodynamic node point in the model.

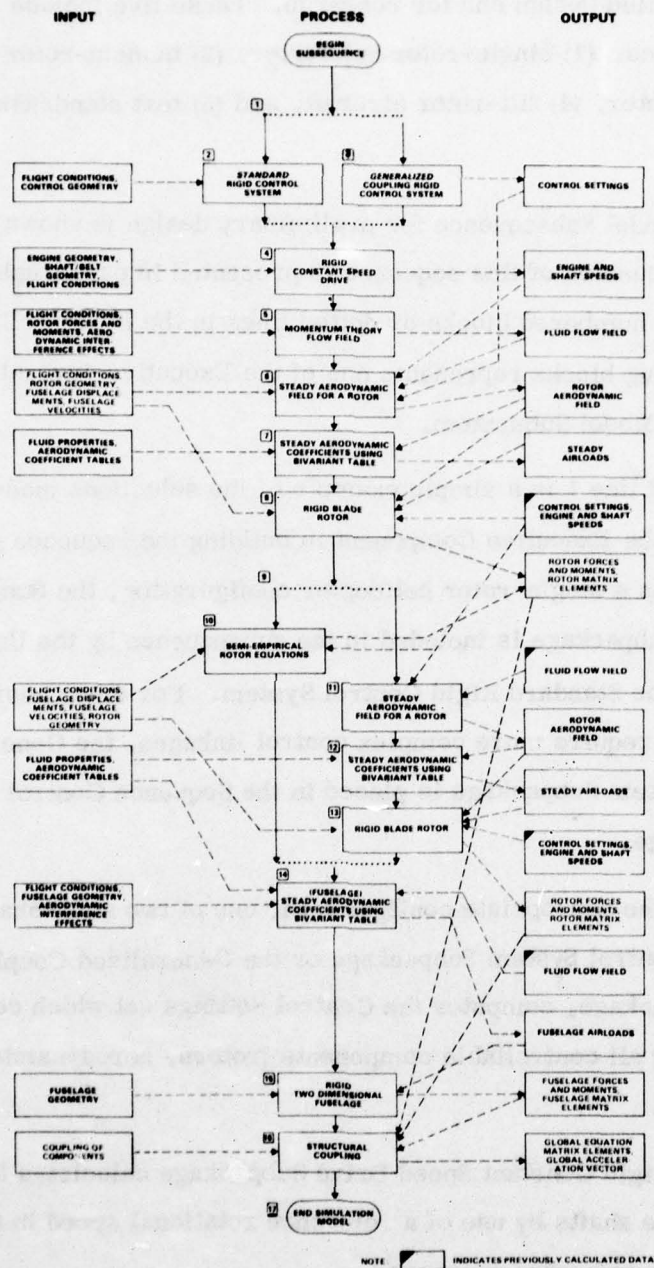


Figure 33. Simulation Model Subsequence for Preliminary Design for All Configurations

6. Blocks 6 through 8 represent one common sub-subsequence which performs a complete rotor analysis including airloads and dynamic response, in this case for the main or number 1 rotor. The Steady Aerodynamic Field for a Rotor Subpackage calculates the total velocity components of the air relative to the blade at each aerodynamic node on the rotor.

7. In block 7 the aerodynamic field is used to find aerodynamic coefficients and to calculate airloads.

8. The Rigid Blade Rotor Subpackage evaluates the forcing function on the rotor and calculates the time-variant coefficients of the mass, damping, and stiffness matrices representing the rotor.

9. Dotted line 9 indicates another choice made indirectly by the user and directly by the User Input Package in constructing the Sequence Control Table. For the single rotor helicopter, the antitorque force is supplied by the simple and fast Semi-Empirical Rotor Equations Subpackage. For configurations with two lifting rotors, the sub-subsequence shown in blocks 11 through 13 is used for the rotor analysis.

10. The Semi-Empirical Rotor Equations Subpackage calculates Rotor Forces and Moments for an antitorque tail rotor.

11, 12, 13. Blocks 11 through 13 perform the same function for rotor 2 as blocks 6 through 8 perform for rotor 1.

14. The Steady Aerodynamic Coefficients Using Bivariant Table Subpackage evaluates Fuselage Airloads as a function of angle of attack and sideslip. The word "fuselage" at the top of this block indicates the aircraft component but is not part of the subpackage name.

15. Block 15 calculates the total Fuselage Forces and Moments and the elements of the mass, damping, stiffness matrices representing the fuselage.



16. The Structural Coupling Package uses the previously generated forcing functions and mass, damping, and stiffness matrices for each configuration component to generate the accelerations and the mass, damping, and stiffness matrices for the coupled configuration.

#### 2.5.2.3 A Simulation Model Subsequence for Detail Design

The tandem-rotor helicopter configuration serves as the example of a Simulation Model Subsequence for detailed design. This subsequence, shown in Figure 34, is discussed below in paragraphs numbered to correspond to the numbered blocks in the figure.

1. The Automatic Flight Control System Subpackage calculates, as a function of stick or actuator motions, the forcing function and the mass, damping, and stiffness matrices representing the control system. This analysis may be sensitive to airspeed or aircraft angular motions as well as other flight conditions.

2. The Generalized Coupling Rigid Control System Subpackage calculates Control Settings for all rotors, surfaces, and other components based on the Controls Matrix Elements from the automatic flight control system and the primary and secondary cockpit control positions. If a user desires to have an elastic dynamic control system representation, he/she need only supply the necessary descriptive input data, and the User Input Package automatically inserts the appropriate System Commands in place of block 2 in this subsequence.

3. The Detail Engine Analysis Subpackage calculates engine performance and the elements of the mass, damping, and stiffness matrices representing the transient response for the engines.

4. The Table Look-Up Flow Field Subpackage uses a three- or four-dimensional table to find rapidly the air velocity vectors at every aerodynamic node point on the aircraft or its environment. This tabular representation of the flow

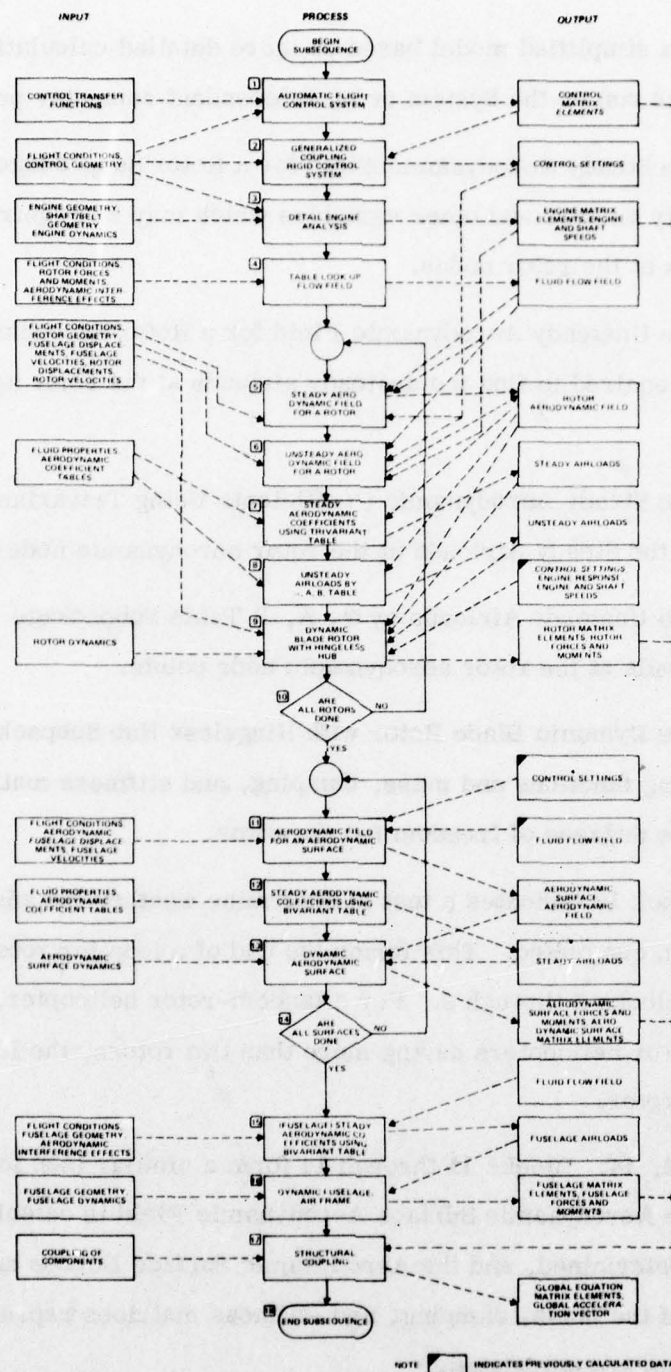


Figure 34. Simulation Model Subsequence for Detailed Design of a Tandem-Rotor Helicopter

field would be a simplified model based on more detailed calculations made during another analysis run on the System or an independent computer program.

5. The Steady Aerodynamic Field for a Rotor Subpackage calculates the relative velocity vectors and other variables which may be required to find the steady airloads at the rotor nodes.

6. The Unsteady Aerodynamic Field for a Rotor Subpackage calculates the variables required to find the unsteady airloads at the rotor aerodynamic node points.

7. The Steady Aerodynamic Coefficients Using Trivariant Table Subpackage calculates the Steady Airloads at the rotor aerodynamic node points.

8. The Unsteady Airloads by  $\alpha$ , A, B Table Subpackage calculates the Unsteady Airloads at the rotor aerodynamic node points.

9. The Dynamic Blade Rotor with Hingeless Hub Subpackage calculates all of the forcing functions and mass, damping, and stiffness matrix coefficients for the analysis degrees of freedom for the rotor.

10. Block 10 indicates a test to determine whether the analysis for all rotors has been completed. This forms the end of a loop for rotor analysis which encompasses blocks 5 through 9. For a tandem-rotor helicopter, this loop is executed twice. For helicopters having more than two rotors, the loop is executed once for each rotor.

11, 12, 13, 14. Blocks 11 through 14 form a similar loop for aerodynamic surfaces. The Aerodynamic Surface Aerodynamic Field is calculated, the Steady Airloads are determined, and the Aerodynamic Surface Forces and Moments and the elements of the mass, damping, and stiffness matrices representing the aerodynamic surface are calculated.

15. The Fuselage Airloads are calculated by the Steady Aerodynamic Coefficients Using Bivariant Table Subpackage.



16. The forcing function and the elements of the mass, damping, and stiffness matrices representing the fuselage are calculated based on the Fuselage Airloads and the dynamic coupling of the fuselage to each of the rotors and aerodynamic surfaces.

17. The Structural Coupling Package uses the previously generated forcing functions and mass, damping, and stiffness matrices for each configuration component to generate the accelerations and the mass, damping, and stiffness matrices for the coupled configuration.

### SECTION 3 - SYSTEM CAPABILITY

The System's analysis capability to accurately predict helicopter performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability for a variety of aircraft configurations is needed during three aircraft life cycle phases: preliminary design, detailed design, and research. In addition, the System must also provide functional capabilities which are secondary to the primary analysis function of the System. These secondary functional capabilities are necessary to ensure that the System is general-purpose, user-oriented, transportable, extendable, maintainable, and efficient. These secondary functional capabilities are identified as executive/support capabilities in the remainder of this section.

Two releases of the System are planned during the Development Phase: the First Level Release and the Second Level Release. The functional requirements to be satisfied by each release are specified in the Baseline Type A System Specification. CSC and BHT have synthesized a design which defines a software system which satisfies the requirements of both System releases and establishes a firm foundation on which future capabilities can be built.

Table 2 identifies the System release which provides the required analysis capabilities for each required aircraft life cycle phase. The detailed capabilities planned for inclusion in the First Level Release and the Second Level Release are identified in Sections 3.1 and 3.2, respectively. For each release, the detailed analysis capabilities are presented first, followed by the detailed executive/support capabilities.

#### **3.1 FIRST LEVEL RELEASE**

As indicated in Table 2, the First Level Release of the System provides a complete set of capabilities for the preliminary design and detailed design aircraft

**Table 2. Achievement of Aircraft Technical Characteristic and Life Cycle Phase Analysis Capabilities in the First and Second Level System Releases**

<div> <div>LIFE CYCLE PHASE</div> <div>TECHNICAL CHARACTERISTICS</div> </div>	PRELIMINARY DESIGN	DETAILED DESIGN	RESEARCH
PERFORMANCE	1	1	2
STABILITY AND CONTROL	1	1	2
LOADS AND VIBRATIONS			
ROTOR LOADS	1	1	2
AIRFRAME LOADS	2	2	2
ENGINE/DRIVE SYSTEM LOADS	2	2	2
CONTROL-SYSTEM/PILOT LOADS	2	2	2
ACOUSTICS	1	2	2
AEROELASTIC STABILITY	1	1	2



life cycle phases to analyze the following technical characteristics of a rotary-wing configuration: performance, stability and control, rotor loads and vibrations, and aeroelastic stability. The First Level Release also provides a complete preliminary design acoustics analysis capability and an extensive set of executive/support capabilities which ensure that the First Level Release of the System is general-purpose, user-oriented, transportable, extendable, maintainable, and efficient. The principal differences between the executive/support capabilities in the First Level Release and those in the Second Level Release are the computers on which the System is available, the amount of interactive interface, and the scope of cost assessment. The First Level Release is planned to be initially available on IBM S/370 and S/360 computers. However, the design makes possible the availability of the First Level Release capability on CDC 6000 and CYBER series of computers 4 months after its availability on the IBM S/370 and S/360 computers. The Second Level Release is planned to be available on both computer families.

The First Level Release has an interactive capability permitting the user to prepare input data and to inspect analysis results at an interactive terminal; the Second Level Release provides, in addition, an interactive tutorial capability and an interactive analysis capability. Finally, the First Level Release provides cost estimates at the conclusion of an analysis run, whereas the Second Level Release also provides cost estimates before execution of an analysis.

Section 3.1.1 presents the detailed analysis capabilities provided by the First Level Release. Section 3.1.2 presents the detailed executive/support capabilities provided by the First Level Release.

#### **3.1.1 First Level Release Analysis Capabilities**

To provide preliminary design and detailed design analysis capabilities for performance, stability and control, rotor loads and vibrations, aeroelastic stability, and acoustics (preliminary design only) in the First Level Release, the System

must be able to generate and use finite element representations of various physical components and to analyze an arbitrarily coupled configuration of physical components.

The First Level Release provides the capability to generate finite element representations of rotors, generalized rigid control systems, engine/drive systems, airframes, an airmass, ground/deck surfaces, and test stands. Three types of rotor representations are included: semiempirical equation representations, rigid blade equation representations, and dynamic analysis representations (with lag dampers, flapping stops, and lag stops) for all hub types. The engine/drive system representations include both rigid and static elastic representations using engine performance tables. Airframe component representations include a rigid fuselage, aerodynamic surfaces, stores, pylons, and a simple landing gear. The airmass representations include steady aerodynamics using tables, unsteady aerodynamics using Theodorsen/Loewy or  $\alpha$ , A, B methods, momentum theory flow fields with or without time delay, and prescribed rotor wakes. The First Level Release also includes a general, systematic, and accurate method for coupling rotating and nonrotating components which employs a minimum number of degrees of freedom, which permits individual components to be independently analyzed, and which is compatible with results generated by the test procedures used throughout the helicopter industry. The specific analysis capabilities to be included in the First Level Release are identified in subsequent subsections in terms of the analysis capabilities provided by each of 10 subsystems of the Technology Component.

#### 3.1.1.1 Simulation Model Initialization

The Simulation Model Initialization Subsystem establishes the local (i.e., component) and global (i.e., configuration) coordinate systems and generates both the initial vector of forcing functions and the constant coefficients of each component's mass, damping, and stiffness matrices. For the First Level Release, the Simulation Model Initialization Subsystem provides the capability to define components

in component local coordinate systems, define the coordinate system of the coupled configuration, assemble the initial vector of forcing functions, and generate the constant coefficients in the mass, damping, and stiffness matrices for the component representations identified in Section 3.1.1. In addition, all user-supplied input data are checked for reasonableness and for potential effects on numerical stability. Specific First Level Release analysis capabilities provided by the Simulation Model Initialization Subsystem are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Calculate constant coefficients for the rotor, the airframe, the engine/drive system, and generalized rigid control system models	Combine Aircraft Components Package
Calculate constant coefficients for the airmass, test stand, and ground/deck models	Combine Environment Components Package
Combine the aircraft and environment models and verify the compatibility of the aircraft and environment models	Combine Aircraft and Environment Components Package
Set up coordinate systems and transformations for all components	Coordinate Systems and Transformations Package
Assemble mass, damping, and stiffness matrices for a rotor in terms of node point displacements	Rotor Finite Element Initialization Package
Compute the natural frequencies and the normalized mode shapes for a rotor	Rotor Modes Package
Compute initial estimates of the rotor wake geometry and the wake element influence coefficients	Wake Initialization Package

### 3.1.1.2 Simulation Modeling

The Simulation Model Subsystem uses the computed results of the Simulation Model Initialization Subsystem to calculate forcing functions and coupling



coefficients for the equations of motion, the distributed aerodynamic loads and vibrations data, and the aerodynamic forces and moments on each aircraft component. The specific simulation modeling capabilities to be provided in the First Level Release are presented in terms of each analysis component. This subsystem also provides, in the First Level Release, the component coupling capability of the System.

### 3.1.1.2.1 Rotor

The rotor simulation modeling capabilities to be included in the First Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Compute rotor hub forces and moments by interpolation from a table	Rotor Map Subpackage
Compute approximate rotor performance using simple equations	Semi-Empirical Rotor Equations Subpackage
Compute approximate ducted-fan performance using simple equations	Semi-Empirical Ducted Fan Subpackage
Calculate the matrix elements and forcing function representing an inelastic rotor	Rigid Blade Rotor Subpackage
Calculate the matrix elements and forcing function representing an elastic rotor with a teetering or gimbaled hub	Dynamic Blade Rotor with Teetering/Gimbaled Hub Subpackage
Calculate the matrix elements and forcing function representing an elastic rotor with an articulated hub	Dynamic Blade Rotor with Articulated Hub Subpackage
Calculate the matrix elements and forcing function representing an elastic rotor with a hingeless hub	Dynamic Blade Rotor with Hingeless Hub Subpackage
Calculate the matrix elements and forcing function representing an elastic rotor with a bearingless hub	Dynamic Blade Rotor with Bearingless Hub Subpackage

Capability	Software Element
Compute rotor shears, moments, accelerations, velocities, and displacements from the calculated response	Rotor Loads and Vibrations Subpackage
Compute for each aerodynamic node point the relative velocity components of the air with respect to the rotor, the angle of attack, the yawed flow angle, the Mach number, and the Reynolds number	Steady Aerodynamic Field for a Rotor Subpackage
Compute for each aerodynamic node point the first and second time derivatives of the angle of attack	Unsteady Aerodynamic Field for a Rotor Subpackage
Compute the loads imparted to the rotor by a viscous or hydraulic lag damper	Viscous/Hydraulic Lag Damper Subpackage
Compute the loads imparted to the rotor by an elastomeric lag damper	Elastomeric Lag Damper Subpackage
Compute the loads transmitted to the mast, hub, and/or blade by flapping stop contact	Rotor Flapping Stops Subpackage
Compute the loads transmitted to the mast, hub, and/or blade by lag stop contact	Rotor Lag Stops Subpackage

#### 3.1.1.2.2 Control System/Pilot

The control system/pilot simulation modeling capabilities to be included in the First Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Provide all riggings for a standard single rotor helicopter control system	Standard Helicopter Rigid Control System Subpackage

Capability	Software Element
Provide all riggings for primary cockpit controls of unconventional rotorcraft (tandem, tilt-rotor, etc.)	Generalized Coupling Rigid Control System Subpackage
Activate secondary cockpit controls	Auxiliary Controls Subpackage

#### 3.1.1.2.3 Engine/Drive System

The engine/drive system simulation modeling capabilities to be included in the First Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Compute the rotational speeds and gear-box losses for all components of the drive system	Rigid Drive System/Constant Speed Analysis Subpackage
Compute engine performance parameters using a table lookup procedure	Engine Performance Table Subpackage
Calculate the stiffness matrix elements representing flexible, but not dynamic, drive shaft	Static Elastic Drive Shaft Subpackage

#### 3.1.1.2.4 Airframe

The airframe simulation modeling capabilities to be included in the First Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Define user-specified test stand motions	Prescribed Motion Test Stand Subpackage
Calculate the matrix elements and the applied external forcing functions representing a fuselage with three rigid-body degrees of freedom	Rigid Two-dimensional Fuselage Subpackage



Capability	Software Element
Calculate the matrix elements and the applied external forcing functions representing a fuselage with six rigid-body degrees of freedom	Rigid Three-dimensional Subpackage
Calculate the matrix elements and forcing function representing a rigid pylon	Rigid Pylon Subpackage
Compute the velocity components, angle of attack, sideslip angle, Mach number, and Reynolds number for an aerodynamic surface	Aerodynamic Field for an Aerodynamic Surface Subpackage
Calculate mass matrix elements representing rigid aerodynamic surfaces and compute the forces and moments transmitted from the surfaces to the airframe	Rigid Aerodynamic Surface Subpackage
Calculate the stiffness matrix elements and forcing function representing a flexible nondynamic landing gear	Simple Landing Gear Subpackage
Calculate the matrix elements and the aerodynamic forcing function representing rigid stores which are either fuselage-mounted or wing-mounted	Rigid Stores Subpackage
Calculate the mass matrix elements and the forcing function representing rigid-body internal cargo	Internal Cargo Subpackage

#### 3.1.1.2.5 Airmass

The airmass simulation modeling capabilities to be included in the First Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Compute aerodynamic coefficients using simple equations	Steady Aerodynamic Coefficients Using Simple Equations Subpackage

Capability	Software Element
Extract aerodynamic coefficients from a two-dimensional table	Steady Aerodynamic Coefficients Using Bivariant Table Subpackage
Extract aerodynamic coefficients from a three-dimensional table	Steady Aerodynamic Coefficients Using Trivariant Table Subpackage
Extract aerodynamic coefficients from a four-dimensional table	Steady Aerodynamic Coefficients Using Quadrivariant Table Subpackage
Compute unsteady aerodynamic coefficients and airloads using the combined theories of Theodorsen and Loewy	Unsteady Airloads by Theodorsen/Loewy Theory Subpackage
Extract unsteady aerodynamic coefficients from the ( $\alpha$ , A, B) table and compute the unsteady airloads	Unsteady Airloads by $\alpha$ , A, B Table Subpackage
Compute the induced velocity distribution on an aerodynamic surface, neglecting unsteady effects	Momentum Theory Flow Field Subpackage
Compute the induced velocity distribution on an aerodynamic surface, including first-order unsteady effects	Momentum Theory Flow Field with Time Delay Subpackage
Compute the induced velocity field due to a prescribed wake geometry	Prescribed Wake Subpackage
Compute the modifications to airfoil section properties due to flap extension	Flap Aerodynamic Coefficients Subpackage
Compute the modifications to airfoil section properties due to spoiler extension	Spoiler Aerodynamic Coefficients Subpackage

#### 3.1.1.2.6 Ground/Deck Surface

The ground/deck simulation modeling capabilities to be included in the First Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Compute the relative motion between the aircraft and a planar, rigid ground/deck	Prescribed Motion Ground/Deck Surface Subpackage

Capability	Software Element
surface undergoing arbitrary user-prescribed motion and contact-dependent forces	
Compute the relative motion between the aircraft and a rigid, two-dimensional ground/deck surface undergoing simple user-prescribed motion and contact-dependent forces	Two-dimensional Ground/Deck Surface Subpackage
Compute the relative motion between the aircraft and a rigid, three-dimensional ground/deck surface undergoing simple user-prescribed motion and contact-dependent forces	Three-dimensional Ground/Deck Surface Subpackage

#### 3.1.1.2.7 Coupling of Components

A capability to couple independently defined configuration components is included in the First Level Release. The component coupling capability transforms the individual component forcing functions and component mass, damping, and stiffness matrices into a single forcing function and a single mass, damping, and stiffness matrix representing the coupled configuration. The mathematical technique employed is defined in Section 2.1 of this report. The component coupling capability is provided by the Structural Coupling Package in the Simulation Model Subsystem. This software element also calculates the global acceleration vector based on the coupled configuration and derives from it the local acceleration vector for each component.

#### 3.1.1.3 Trim

Two alternative trim solution capabilities will be provided in the First Level Release. These capabilities, together with the software elements which provide them, are listed below.



Capability	Software Element
Iterate all trim equations to find a steady-state condition	Simultaneous Iterate-to-Trim Package
Use autopilot to simulate flight to a steady-state condition	Fly-to-Trim Package

#### 3.1.1.4 Maneuvers

Two alternative capabilities are provided in the First Level Release to calculate control motion disturbances for transient solutions. These capabilities, together with the software elements which provide them, are listed below.

Capability	Software Element
Calculate time-dependent control positions from user-specified control motions	Prescribed Control Motions Package
Calculate control motions required to follow a user-specified aircraft response	Prescribed Aircraft Response Package

#### 3.1.1.5 Stability and Control

The complete stability and control analysis capability is included in the First Level Release. The detailed capabilities needed to calculate stability and control data are listed below, together with the software elements which provide the detailed capabilities.

Capability	Software Element
Calculate stability derivatives and generate linearized equations of motion	Linearized Equations for Stability and Control Package
Calculate eigenvalues and eigenvectors for stability and control	Stability Eigenvalues and Eigenvectors Package
Calculate stability and control transfer functions and frequency response for selected control inputs	Transfer Functions and Frequency Response Package

### 3.1.1.6 Acoustics

A capability to calculate an empirical noise field, including the effects of noise dissipation and noise reflection, is provided by the Sound Propagation Package in the First Level Release.

### 3.1.1.7 Aeroelastic Stability

Three alternative aeroelastic stability analysis capabilities will be provided in the First Level Release. These capabilities are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Calculate stability frequencies and damping coefficients by eigenanalysis of linearized equations	Linear Aeroelastic Stability Analysis Package
Calculate stability frequencies and damping coefficients for equations with periodic coefficients	Floquet Analysis Package
Calculate stability frequencies and damping coefficients by processing time history data	Aeroelastic Stability Postprocessing Package

### 3.1.1.8 General Mathematical Operations

All general mathematical operation capabilities needed to support First Level Release and Second Level Release analysis capabilities are provided as part of the First Level Release. All included matrix manipulation capabilities accept matrices which exceed the size of computer memory available to the System (i.e., perform operations on data residing in whole or in part on peripheral storage). The general mathematical operation capabilities are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Solve a set of linear equations with real or complex coefficients	Linear Algebraic Equation Solution Package

Capability	Software Element
Solve a set of nonlinear, first-order, stiff, or nonstiff ordinary differential equations	Differential Equation Solution Package
Determine the eigenvalues and eigenvectors for real general, complex general, complex Hermitian, and real symmetric matrices	Eigenvalue/Eigenvector Package
Compute the first and second derivatives of a tabular function	Numerical Differentiation Package
Multiply and/or add two real or complex matrices	Matrix Multiplication and Addition Package
Perform a general harmonic analysis of a given time history	Harmonic Analysis Package
Invert square, nonsingular, real or complex matrices	Matrix Inversion Package
Compute the frequency content and associated damping of a given time history by using a moving-block Fast Fourier Transform	Moving-Block Fast Fourier Transform Package
Transform a three-component vector from one orthogonal coordinate system to another	General Coordinate Transformation Package
Compute autocorrelation and cross-correlation functions for given time histories	Statistical Functions Package
Decompose nonsingular, square matrices	Matrix Decomposition Package
Perform a linear, cubic, or bicubic interpolation or extrapolation of a function	Interpolation/Extrapolation Package
Solve sets of nonlinear equations	Nonlinear Algebraic Equations Solution Package
Integrate a real function over a closed interval	Quadrature Package



Capability	Software Element
Compute the frequency content and associated damping of a given time history by using Prony's Method	Prony's Method Harmonic Curve Fit Package
3.1.1.9 External Models Interface	
One software element (External Model 1 Package) is included in the First Level Release to provide an interface from the System to one Government-specified external model. The specific external model has not yet been specified.	
3.1.1.10 Accuracy Assessment	
No accuracy assessment capability is planned for the First Level Release.	
3.1.2 <u>First Level Release Executive/Support Capabilities</u>	
Virtually all executive/support capabilities specified for inclusion in the Second Level Release are included in the First Level Release as well. There are five categories of executive/support capabilities included in both releases: user interface, run-time management, data base management, operating system services, and support services.	
3.1.2.1 User Interface	
The user interface capabilities planned for inclusion in the First Level Release will facilitate the use of the System by both engineering users and methods developers. The following is a list of user interface capabilities to be included in the First Level Release:	
<ol style="list-style-type: none"> <li>1. An interface between the engineering user and the System is defined which allows the user to analyze a problem without requiring the user to understand the software design of the System.</li> <li>2. The user does not need to specify the sequence of execution for analysis software elements--the execution sequence will be automatically</li> </ol>	

defined by the System based upon the characteristics of the configuration to be analyzed, the type of analysis desired, and the level of detail (i.e., aircraft life cycle phase: preliminary design, detailed design, or research) at which the configuration is to be analyzed.

3. The user can define the configuration to be analyzed in terms of any combination of component models previously stored in a Master Data Base and component models defined by the user at analysis run time.
4. If the user wishes to define a modified version of a component model which resides in a Master Data Base, only the modifications need be specified at analysis run time.
5. A simple high-level System Command Language is available if the user wishes to explicitly control the execution sequence of analysis software elements.
6. A Master Command File, containing validated sets of System Command Sequences needed to use the particular functional capabilities provided by the First Level Release, will be provided to all users of the First Level Release.
7. Frequently used sets of System Commands can be stored by authorized users (either engineering users or methods developers) in a Master Command File for subsequent use.
8. The user may construct a complete sequence of System Commands by specifying any combination of predefined sets of System Command Sequences residing on a Master Command File and command sequences defined by the user at analysis run time.
9. If the user wishes to define a modified version of a System Command Sequence which resides on a Master Command File, only modifications need be specified at analysis run time.

10. User input data can be provided either in the form of a deck of cards or in the form of a card image file prepared at an interactive terminal.
11. Printed reports of all input explicitly supplied by the user are generated.
12. If desired, printed reports of all input implicitly supplied by the user (i.e., component models extracted from a Master Data Base, including all user-specified modifications and System Command Sequences extracted from a Master Command File, including all user-specified modifications) are generated.
13. All user-specified input data are inspected for actual or possible errors. All resulting diagnostic messages are reported using clear and concise engineering-oriented terminology.
14. Component and configuration model definition data can be plotted.
15. Analysis results may be output as printed reports, plots, or combinations thereof.
16. Multiple formats of printed reports and plots are provided so that the user can select the most meaningful format for inspecting analysis results.
17. All reports generated by the System are available for inspection at an interactive terminal.

#### 3.1.2.2 Run-Time Management

The purpose of the run-time management capability is to supervise the processing that takes place during an analysis run. All run-time management capabilities needed for the Second Level Release are planned to be included in the First Level Release as well. Run-time management capabilities are particularly important



from two aspects: efficiency and extendability. Capabilities are provided to minimize the costs of analyzing both small and large problems and to facilitate experimentation with new or modified analysis capabilities. The following is a list of detailed run-time management capabilities to be provided by the First Level Release:

1. Memory, CPU, and device utilization statistics are collected as each analysis software element is executed. (These statistics can be used to assess the performance impact of new or modified analysis capabilities.)
2. Linkage overhead is minimized by grouping related software elements into load-modules. (This permits a direct call between software elements rather than the more costly indirect linkage through either an Executive or operating system service.)
3. Inactive load-modules are not deleted from memory until the memory is needed for other load-modules or for data. (This avoids potentially unnecessary overhead needed to reload load-modules.)
4. All available memory other than the memory required by active load-modules is used, if needed, for the data required by the active load-modules. (This minimizes the amount of I/O overhead incurred by small problems.)
5. Memory-resident data are not output to a peripheral device unless the memory is needed for additional data by another load-module. (This minimizes the amount of I/O overhead incurred by small problems.)
6. If the user wishes to define a new sequence of System Commands that use already existing software elements, no software modifications are necessary and no load-modules need be rebuilt. (The run-time

management function controls the sequence of software element executions based on System Commands, i.e., the sequence of software element executions is not defined in a software element.)

7. A software element can generate System Commands to accomplish its function. (This typically is used when a software element, in order to accomplish its designated function, must use a function provided by a different software element.)
8. Meaningful diagnostics describing internal software problems are provided, along with the corresponding information needed to identify the cause of the problem.
9. An analysis run can be restarted, following an unexpected termination or normal completion, with modified model data or a modified sequence of System Commands. (This avoids unnecessary repetition of analysis processing.)
10. Restart information generated by an analysis run can be used by any subsequent analysis run. (This avoids recalculating information which would be otherwise unaffected by the analysis performed in the subsequent analysis run, e.g., the stiffness, mass, or damping matrix representing a component common to multiple aircraft configurations.)

#### 3.1.2.3 Data Base Management

The complete data base management capability is included in the First Level Release. The data base management capability is specially designed to meet the efficiency and extendability needs of the Second Generation Comprehensive Helicopter Analysis System. The following list summarizes the data base management capabilities provided in the System:

1. Data base storage and retrieval services are provided, which eliminate the need for a software element to consider the location or

organization of the data; i.e., data are requested by name rather than by relative location. (This approach also allows part or all of the data base to be memory resident, depending on problem size, without affecting the logical operation of a software element.)

2. At analysis run initiation, a special run-time data base is formed containing only the Master Data Base information, as augmented by user data, needed to perform the analysis. (This reduces the size of the data base needed for the analysis, thus reducing data access overhead.)
3. Modifications to a Master Data Base and Master Command File are not permitted during an analysis run. (This allows a user installation to establish controlled procedures for updating a Master Data Base and a Master Command File.)
4. Support capabilities are provided to modify a Master Data Base and a Master Command File and to generate output reports describing the contents thereof.
5. Frequently used component model data and System Command Sequences can be stored in a Master Data Base and a Master Command File, respectively. (This permits the inclusion of the component models and command sequences in subsequent analysis runs without respecifying the detailed model data or the System Commands.)

#### 3.1.2.4 Operating System Services

The operating system services provided by the System are designed to facilitate transportability. For the First Level Release, the services will be provided for



IBM S/370 and S/360 computers. The following is a list of operating system service capabilities to be provided for both releases of the System:

1. A computer-independent interface to storage management, file management, program management, and cost assessment/diagnostic support services is provided. (This permits the First Level Release capability to be available on CDC 6000/CYBER computers approximately 4 months after its availability on IBM S/370 and S/360 computers.)
2. Memory utilization, CPU utilization, and file utilization statistics are collected throughout the run and converted to cost estimates at analysis run conclusion.

#### 3.1.2.5 Support Services

A complete set of services is provided in the First Level Release to support methods developers in the development, testing, and documentation of new or modified System software. In addition, services are also provided to support the management of the System software and data configuration and to support the installation of the System on other host computers. A separate set of support services is provided in the First Level Release for IBM S/370 and S/360 computers and CDC 6000 and CYBER computers.

#### 3.2 SECOND LEVEL RELEASE

As indicated in Table 2, the analysis capabilities to be added to the First Level Release capability that results in the Second Level Release capability provide a complete capability to accurately predict helicopter performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability for a variety of aircraft configurations. The complete capability is available for aircraft configurations during three life cycle analysis phases: preliminary design, detailed design, and research. Major new executive/support capabilities provided in the

Second Level Release include a fully interactive user environment and an analysis run cost prediction capability.

### 3.2.1 Second Level Release Analysis Capabilities

The level of detail at which individual components of a physical configuration can be represented is significantly expanded in the Second Level Release. This expansion is essential if the System is to be useful during the research life cycle phase of helicopter analysis. A significant addition to the First Level Release is a capability to assess the effects of damage to, or failure of, the various aircraft components (rotor, fuselage, engine/drive system, controls, etc.). This failure/damage assessment capability is particularly useful during the detailed design and research life cycle phases of helicopter analysis.

The Second Level Release also includes a significantly expanded acoustics analysis capability. Specific capabilities are added to the First Level Release to calculate rotor external noise, engine noise, gearbox noise, and noise contributions from aircraft accessories.

Finally, an automated capability is planned for supporting the very important task of assessing the accuracy and validity of the System analysis capabilities. This accuracy assessment capability will be particularly useful during the System Validation Phase.

#### 3.2.1.1 Simulation Model Initialization

The Simulation Model Initialization Subsystem software elements included in the First Level Release are upgraded to reflect model initialization requirements which correspond to the new simulation modeling capabilities included in the Second Level Release. No new software elements, beyond those provided in the First Level Release, are planned for the Second Level Release.

#### 3.2.1.2 Simulation Modeling

Major additions to the First Level Release simulation modeling capabilities are planned for the Second Level Release. The rotor representations are expanded to

include semiempirical circulation control rotors, semiempirical reaction drive rotors, pendulum absorbers, control load reduction devices, and servo flaps. The control system/pilot representations are expanded to include elastic control systems, dynamic control systems, force feel systems, automatic flight control systems, control feedback from force/motion sensors, and pilot transfer functions. The engine/drive system representations are expanded to include a detailed engine analysis with governor and fuel control; a rigid, static elastic, and dynamic gearbox; a dynamic drive/shaft; and a clutch. The airframe representations are expanded to include a static elastic and dynamic fuselage, a static elastic and dynamic aerodynamic surface, vibration control devices, suspended cargo, complex landing gear, dynamic stores, and hoist and load stabilization devices. The airmass representations are expanded to include free rotor wake, cable aerodynamics, wind tunnel corrections, and aerodynamic interference effects between and among rotors, aerodynamic surfaces, and bodies. Also included is an advanced unsteady aerodynamic analysis capability and an aerodynamic solution for arbitrary bodies and nonrotating lifting surfaces. Representations of other analysis components have been expanded to include a dynamic test stand, an elastic or plastic deformable ground or deck surface, and a water surface. For each aircraft component (i.e., rotor, control system, engine/drive system, and airframe), the capability to simulate component failure or damage is also added in the Second Level Release. The specific simulation modeling capabilities to be added to the First Level Release are presented in Sections 3.2.1.2.1 through 3.2.1.2.7 in terms of the various analysis components.

#### **3.2.1.2.1 Rotor**

The additional rotor simulation modeling capabilities to be included in the Second Level Release are listed below, together with the software elements which provide the capabilities.



Capability	Software Element
Calculate the matrix elements and forcing function representing steady-state, oscillatory, and transient rotor response, including elastic bending and torsion of the hub and blade components	Elastic Substructured Rotor Analysis Subpackage
Compute approximate performance of a circulation control rotor using simple equations	Semi-Empirical Circulation Control Rotor Subpackage
Compute approximate performance of a reaction drive rotor using simple equations	Semi-Empirical Reaction Drive Rotor Subpackage
Calculate the forcing function representing the shears and moments input to the rotor from a reaction drive	Reactive Drive Rotor Subpackage
Calculate the matrix elements representing a blade-mounted out-of-plane pendulum	Rotor Out-of-Plane Pendulum Subpackage
Calculate the matrix elements representing a blade-mounted in-plane pendulum	Rotor In-Plane Pendulum Subpackage
Calculate the matrix elements representing a rotor-mounted control load reduction device, and the forcing function representing the loads transmitted to the rotor	Rotor Control Load Reduction Devices Subpackage
Calculate the matrix elements representing a rotor servo-flap, and the forcing function representing the loads imparted to the rotor and control system by the flap	Rotor Servo Flaps Subpackage

#### 3.2.1.2.2 Control System/Pilot

The additional control system/pilot simulation modeling capabilities to be included in the Second Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Calculate the stiffness matrix elements and the forcing function representing a flexible, but not dynamic, control system	Static Elastic Control System Subpackage
Calculate the matrix elements and forcing function for a dynamic control system	Dynamic Control System Subpackage
Compute loads and vibrations for all dynamic components of the control system from the response calculated for those components	Control System Loads and Vibrations Subpackage
Calculate the matrix elements and control motion forcing functions representing an Automatic Flight Control System	Automatic Flight Control System Subpackage
Calculate the matrix elements and control motion forcing function representing control feedback from force/motion sensors	Control Feedback from Force/Motion Sensors Subpackage
Calculate the matrix elements and control motion forcing function representing an artificial force feel system	Force Feel System Subpackage
Calculate the matrix elements and forcing function representing pilot response to vibrations, aircraft motions, and aircraft accelerations	Pilot Transfer Function Subpackage
Compute control positions necessary for the operation of a circulation control rotor	Simple Circulation Control Rotor Control System Subpackage

### 3.2.1.2.3 Engine/Drive System

The additional engine/drive system simulation modeling capabilities to be included in the Second Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Generate the matrix elements and forcing function representing an engine and its transfer functions using data provided by engine manufacturers	Engine Manufacturer Simulation Subpackage
Calculate the matrix elements and forcing function representing engine performance characteristics and engine dynamics	Detail Engine Analysis Subpackage
Calculate the matrix elements representing the steady-state, transient, and dynamic characteristics of all engine management components	Engine Governor and Fuel Control Subpackage
Calculate the matrix elements and the forcing functions for the engine and for the engine management components related to a reaction drive mechanism	Reaction Drive Subpackage
Calculate the matrix elements and the forcing functions for the engine and for the engine management components related to a circulation control drive mechanism	Circulation Control Drive Subpackage
Calculate the matrix elements and forcing functions for the engine and for the engine management components related to turbojet and turbofan engines	Auxiliary Propulsion Subpackage
Calculate the matrix elements and the forcing function representing a dynamic drive shaft.	Dynamic Torsion and Bending Drive Shaft Subpackage
Calculate the mass matrix elements representing a rigid gearbox	Rigid Gearbox Subpackage
Calculate the elements of a stiffness matrix representing a static elastic gearbox with gear train	Static Elastic Torsion Gearbox Subpackage
Calculate the matrix elements representing the torsional dynamic characteristics of a gearbox with gear train	Dynamic Torsion Gearbox Subpackage



Capability	Software Element
Calculate the stiffness matrix elements representing a flexible nondynamic drive belt	Static Elastic Drive Belt Subpackage
Calculate the matrix elements representing a clutch component	Clutch Analysis Subpackage
Calculate the loads and vibrations for each drive system component based upon the response of each drive system component	Drive System Loads and Vibrations Subpackage
Calculate the matrix elements representing a dynamic torsional drive belt-pulley system	Dynamic Torsion Drive Belt Subpackage

#### 3.2.1.2.4 Airframe

The additional airframe simulation modeling capabilities to be included in the Second Level Release are listed below, with the software elements which provide the capabilities.

Capability	Software Element
Calculate the matrix elements and forcing function representing a flexible test stand support structure	Dynamic Test Stand Subpackage
Calculate both the matrix elements representing a fuselage or a complete airframe structure, and a forcing function representing applied external forces	Dynamic Fuselage/Airframe Subpackage
Calculate the stiffness matrix elements and the forcing function representing a static elastic pylon	Static Elastic Pylon Subpackage
Calculate the matrix elements and the forcing function representing a dynamic pylon	Dynamic Pylon Subpackage
Calculate the stiffness matrix elements and the aerodynamic forcing function representing static elastic aerodynamic surfaces	Static Elastic Aerodynamic Surface Subpackage

Capability	Software Element
Calculate the matrix elements and the aerodynamic forcing function representing dynamic aerodynamic surfaces	Dynamic Aerodynamic Surface Subpackage
Calculate the matrix elements and the forcing function representing wheeled or skid landing gear	Detailed Landing Gear Subpackage
Calculate the mass matrix elements and the aerodynamic forcing function representing externally suspended rigid-body cargo	Rigid Suspended Cargo Subpackage
Calculate the matrix elements and the aerodynamic forcing function representing externally suspended flexible cargo	Dynamic Suspended Cargo Subpackage
Calculate the matrix elements and the forcing function representing a flexible cable	Cable Subpackage
Calculate the matrix elements and the forcing function representing either passive or active vibration control devices	Vibration Control Device Subpackage
Calculate the matrix elements and the forcing function representing cargo stabilization devices attached to the airframe	Suspended Cargo Stabilization Devices Subpackage
Calculate the matrix elements and the forcing function representing a dynamic vibration control device for isolating cargo loads from the fuselage	Hoist and Load Isolation Subpackage
Calculate airframe loads and vibrations at any node point based on the response of each airframe component	Airframe Loads and Vibrations Subpackage
Calculate the matrix elements and forcing function representing dynamic fuselage-mounted and wing-mounted stores	Dynamic Stores Subpackage

<u>Capability</u>	<u>Software Element</u>
Calculate the matrix elements and forcing function representing the effects of fuel and fuel slosh on fuselage dynamics	Fuel Subpackage

#### 3.2.1.2.5 Airmass

The additional airmass simulation modeling capabilities to be included in the Second Level Release are listed below, together with the software elements which provide the capabilities.

<u>Capability</u>	<u>Software Element</u>
Compute unsteady aerodynamic coefficients using state-of-the-art stall delay techniques	Unsteady Airloads for Time Delay Subpackage
Compute the matrix elements and the forcing function representing the geometry of, and the induced velocity field due to, a free wake	Free Wake Subpackage
Determine the induced velocity field at all lifting surfaces from a table of induced velocities	Table Look-Up Flow Field Subpackage
Compute the aerodynamic coupling effects between two rotor systems	Rotor-to-Rotor Interference Subpackage
Compute the aerodynamic coupling effects between a rotor and an aerodynamic surface	Rotor-to-Aerodynamic Surface Interference Subpackage
Compute the aerodynamic coupling effects between a fuselage and an aerodynamic surface	Fuselage-to-Aerodynamic Surface Interference Subpackage
Compute the aerodynamic coupling effects between two airframe components	General-Purpose Aerodynamic Interference Subpackage
Compute the aerodynamic loads on, and velocities due to, arbitrarily shaped, nonrotating, nonlifting bodies and surfaces	Aerodynamic Panel Method for Arbitrary Bodies Subpackage



Capability	Software Element
Compute the aerodynamic loads on, and velocities due to, an arbitrarily shaped, nonrotating, lifting or nonlifting surface	Nonrotating Aerodynamic Surface Potential Flow Subpackage
Compute the forces and moments generated by a circulation control rotor by the use of simple equations or table look-up	Semi-Empirical Circulation Control Aerodynamics Subpackage
Compute the aerodynamic loads on a cable in transverse flow	Cable Aerodynamics Subpackage
Compute analytically the aerodynamic effects of wind tunnel boundaries on a body in the tunnel	Analytical Wind Tunnel Boundary Corrections Subpackage
Compute from empirical data the aerodynamic effects of wind tunnel boundaries on a body in the tunnel	Empirical Wind Tunnel Boundary Corrections Subpackage

#### 3.2.1.2.6 Ground/Deck Surface

The additional ground/deck surface modeling capabilities to be included in the Second Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Calculate the matrix elements and the contact forcing function representing an elastic three-dimensional ground/deck surface	Dynamic Elastic Ground/Deck Surface Subpackage
Calculate the matrix elements and the contact forcing function representing an elastic/plastic three-dimensional ground/deck surface	Elastic/Plastic Ground/Deck Surface Subpackage
Calculate the matrix elements and the contact forcing function representing a simple water surface	Water Surface Subpackage

#### 3.2.1.2.7 Coupling of Components

No additional methods for coupling components, beyond the component modes method included in the First Level Release, are planned for inclusion in the Second Level Release. However, if Development Phase funding permits, it is recommended that the substructure analysis method for coupling components be included in the Second Level Release (see Section 2.1 of this report for a more complete discussion of this recommendation).

#### 3.2.1.3 Trim

A third trim solution capability is added to the two capabilities included in the First Level Release. This new capability finds a steady-state solution by dividing the equations of motions into groups and iterating for a solution on each group separately. This approach eliminates convergence problems experienced when the iteration includes all the equations of motion. This new Second Level Release capability is provided by the Decoupled Iterate-To-Trim Package.

#### 3.2.1.4 Maneuvers

Two new maneuver-related capabilities are added to the maneuver capabilities included in the First Level Release. The first new capability (provided by the Gust Response Package) calculates the disturbance in the fluid flow field due to user-specified gusts, trailing vortices, or weapons blasts. The second new capability (provided by the Initiate Failure/Damage Effects Package) activates the modeling of the effects of component failure or component damage as and when indicated by the user. Actual component failure or damage is simulated by the simulation modeling capabilities provided for each aircraft component (Section 3.2.1.2).

#### 3.2.1.5 Stability and Control

The complete stability and control analysis capability is provided in the First Level Release.

### 3.2.1.6 Acoustics

The additional acoustics analysis capabilities to be included in the Second Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Calculate the rotational component of rotor external noise at integer harmonics	Rotor Rotational Sound Subpackage
Calculate the broadband component of rotor external noise at other than integer harmonics	Rotor Broadband Sound Subpackage
Calculate the reciprocating engine noise component of helicopter far-field and near-field external noise	Reciprocating Engine Sound Subpackage
Calculate the turbine engine noise component of helicopter far-field and near-field external noise	Turbine Engine Sound Subpackage
Calculate the external and internal noise contributions from the main transmission or any gearbox, taking into account the type of gearing, case sound transmissibility, isolation, and installation effects	Gearbox Sound Package
Calculate the external and internal noise contributions from accessories such as oil cooler fans, hydraulic pumps, bypass valves, ECUs, APUs, ventilation fans, generators/alternators, and avionics equipment	Accessories Sound Package

### 3.2.1.7 Aeroelastic Stability

The three alternative aeroelastic stability analysis capabilities included in the First Level Release provide all the required aeroelastic stability analysis capabilities.



### 3.2.1.8 General Mathematical Operations

The general mathematical operation capabilities included in the First Level Release provide all the required general mathematical operation capabilities.

### 3.2.1.9 External Models Interface

No interface capability to other external models (beyond the one external model interface capability included in the First Level Release) is planned for the Second Level Release because no external models are currently specified.

### 3.2.1.10 Accuracy Assessment

The Second Level Release provides automated support for assessing the accuracy of an analysis. This automated capability is intended to support the System Validation Phase. The detailed accuracy assessment capabilities planned for inclusion in the Second Level Release are listed below, together with the software elements which provide the capabilities.

Capability	Software Element
Define a series of cases to be run with perturbations to user-specified input evaluation variables	Set Up Accuracy Assessment Cases Packages
Calculate the partial derivatives of each of the user-specified output evaluation variables with respect to each of the specified input evaluation variables	Compute Sensitivity Factors Package
Find standard deviations for the output variables as a basis for the expected values and reasonable ranges of experimental data	Generate Expected Values and Ranges Package
Compare computed results with experimental results to support the evaluation of the validity and accuracy of System results	Compare Computed Values Versus Experimental Data Package

### **3.2.2 Second Level Release Executive/Support Capabilities**

The major Second Level Release enhancements to the First Level Release executive/support capabilities are in two categories: user interface and operating system services. The detailed executive/support capabilities added to the First Level Release in the Second Level Release are presented in terms of the same five capability categories used in Section 3.1.2 to present the First Level Release executive/support capabilities.

#### **3.2.2.1 User Interface**

The major Second Level Release enhancements to the First Level Release user interface capabilities are in two areas: complete user interactivity and prediction of analysis run costs. Specific capability enhancements to the user interface capabilities provided in the First Level Release are as follows:

1. A tutorial capability is available to users at interactive terminals. (This tutorial capability will help the user define component model data and compose System commands.)
2. The user may suspend processing after any desired System Command in order to inspect analysis results thus far computed, to modify the sequence of System Commands, or to do both.
3. A prediction of the costs of an analysis will be provided before the analysis is performed. (If the predicted costs indicate delayed access to the computer because of installation restrictions, the user can then reduce the complexity of the analysis, and the resulting analysis costs, to gain quicker access to the computer.)
4. The user can define, at an interactive terminal, the formats to be used for each printed report or plot. (The First Level Release provides a similar capability; however, the First Level Release capability is restricted to a predefined set of optional formats. With the

Second Level Release, the user can explicitly define the formats best suited to the user and to the problem being analyzed.)

5. A Master Command File containing an expanded set of validated System Command Sequences needed to use the particular functional capabilities of the Second Level Release is provided to all users of the Second Level Release.

#### 3.2.2.2 Run-Time Management

Run-time management capabilities are complete for the First Level Release and therefore no additional capabilities are planned for the Second Level Release.

#### 3.2.2.3 Data Base Management

Data base management capabilities are complete for the First Level Release and therefore no additional capabilities are planned for the Second Level Release.

#### 3.2.2.4 Operating System Services

For the Second Level Release, all First Level Release operating system service capabilities are available on both IBM S/370 and S/360 computers and CDC 6000 and CYBER computers. One additional operating system service capability is included in the Second Level Release: a computer and device-independent interface to interactive terminals is provided.

#### 3.2.2.5 Support Services

Support service capabilities are complete for the First Level Release and therefore no additional capabilities are planned for the Second Level Release.



## SECTION 4 - SYSTEM USE

The various uses of the System are divided into four categories:

- The use of the System to perform standard analyses
- The use of the System to develop new analysis capabilities
- Interactive use of the System
- Supporting use of the System

Section 4.1 summarizes these four uses. The succeeding sections contain more detailed information about topics referred to in Section 4.1.

### 4.1 SYSTEM USE OVERVIEW

To perform a standard engineering analysis, the System user submits a deck of cards (or a file of card images) containing information in the form described in Section 4.3, User Input. The user input is designed to allow the user to express the analysis to be performed in terms that are descriptive of the problem to be solved rather than in terms of the steps the System should take to solve the problem. Knowledge of the physical configuration to be analyzed is emphasized rather than knowledge of the System that will perform the analysis. The user input is also designed to be self-documenting. Most of the statements that appear in the user input are English-language statements that will be clear to the user with minimal explanation. In addition, the user may supply comment cards to further document the input.

Most of the information required to describe the physical configuration to be analyzed, the conditions for the analysis, and the failure or damage effects to be considered is normally contained in the Master Data Base, described in Section 4.2. If the information is contained in the Master Data Base, the user simply names the sets in the Master Data Base that contain the information. Changes to the data contained in the Master Data Base for this run only may be specified in user input. If the information required to describe the analysis to be performed is not contained

AD-A063 921

COMPUTER SCIENCES CORP SILVER SPRING MD SYSTEM SCIEN--ETC F/G 1/3  
THE PREDESIGN PHASE OF THE SECOND-GENERATION COMPREHENSIVE HELI--ETC(U)  
OCT 78

DAAJ02-77-C-0057

UNCLASSIFIED

USARTL-TR-78-41

NL

3 OF 5  
AD  
A063921



in the Master Data Base, the entire description may be contained in user input. The maintenance and dissemination of the contents of the Master Data Base are the responsibility of each individual installation and will undoubtedly be performed in many different ways. Typically, a catalog of the current sets in the Master Data Base will be maintained either by distribution to the System users or in a file accessible to System users. To make the most efficient use of the System, a user must know what data are available in the Master Data Base.

The System action upon receipt of user input is to create a Run Data Base, described in Section 4.6, and a Sequence Control Table. The Run Data Base contains input data from the Master Data Base as modified by user input and, as the run proceeds, it will contain intermediate and final results. The Sequence Control Table contains System Commands, described in Section 4.5, selected from the Master Command File based on information implicit in user input.

To use the System for developing new analysis capabilities, a more detailed knowledge of the System is required than for performing standard analyses as described above. The methods developer will normally be adding new software elements to the System and will be testing their effectiveness in performing helicopter analyses. To perform a test, the methods developer must use the Support Complex to produce one or more load-modules containing the software elements to be added. If new data elements are required in the Master Data Base or the Run Data Base, they must be defined by use of the Support Complex. The methods developer must then create a sequence of System Commands that will cause the new software elements to be executed in their proper context. The form of user input as described in Section 4.3 will be extended by the addition of a seventh section, which will contain explicit definition of System Command Sequences either by modification of existing sequences in the Master Command File or by creation of completely new sequences. By use of user input containing information in the seventh section, the user developing new capability may create any sequence of System Commands required.



The System can be used interactively to perform any of the functions described above, because the host operating systems for both the Host 1 and the Host 2 families of computers contain the capability to interactively create files, submit jobs, and examine files. This capability can be used to create a file containing user input, to submit a job to perform the analysis specified in the user input file, and to examine files containing the results of the analysis. The interactive capabilities of the System, in addition to those supplied by the host operating systems, are as follows:

1. Print or plot information using the System's formatting capability to produce reports specifically designed for the engineering user of the System. This capability may be used to examine input data, intermediate results, or final results.
2. Either modify an existing sequence of System Commands or create a new sequence of System Commands, and then execute the sequence of System Commands. This capability may be used to test partial sequences of System Commands. When System Commands are executed from an interactive terminal, the execution of a STOP command causes control to return to the user at the terminal. The terminal user may then examine intermediate results, continue execution at the command immediately following the STOP command, further modify the sequence of System Commands, continue execution from any other point in the sequence, or perform any other function normally available from an interactive terminal. Specifically, to monitor the result of executing each System Command in a sequence, the terminal user would insert a STOP command after each System Command in the sequence and execute the sequence. The terminal user then regains control at the terminal after each System Command is executed and may examine the results.

3. Provide tutorial information. Tutorial information includes general information about the System and its use, information about the current contents of the Master Data Base or the Run Data Base, and information about sequences of System Commands. Tutorial information is designed to be used by the engineering user to prepare input for an analysis run or by the methods developer to prepare runs for testing new or improved analysis methods.

The System may be used to support its engineering analysis use and methods development use in several ways. The most frequent supporting use of the System will be for maintenance of the Master Data Base. The Master Data Base may be created or changed only by an authorized person at an installation through use of the Data Base Support Package of the Support Complex. The conventions for set names, criteria for inclusion of data, authorization of update capability, and other policy matters regarding the maintenance of the Master Data Base will be decided individually at each installation. The goals of the Master Data Base maintenance policy should be to

1. Maintain a stable, verified source of data
2. Quickly add data likely to be used repeatedly
3. Rapidly disseminate knowledge of Master Data Base contents to all System users

The capabilities of the System can be used to accomplish the first two of these goals. Test versions of the Master Data Base may exist simultaneously with the primary Master Data Base. This capability allows the verification of data prior to inclusion in the primary Master Data Base and allows the test of new software elements requiring new data elements.

The System will assist in making efficient use of the System; i. e., the System has a cost assessment capability that can be used to estimate the cost of an analysis

run without actually making the run. By means of this capability, the System user can decide whether the results to be derived from an analysis run are worth the cost of the run or can choose the least costly of several alternative ways to achieve the desired results.

If a directly created System Command Sequence is used repeatedly at an installation, it can be made more easily available to System users by the use of another supporting capability. The new System Command Sequence may be added to the Master Command File by use of the Data Base Support Package of the Support Complex.

In summary, the System may be used in a number of different ways and for a number of different purposes. The primary purpose of the System, to perform helicopter analyses, may be accomplished in a manner ranging from a simple description of the physical configuration to be analyzed to a complete description of the steps to be followed by the System in performing the analysis. Either batch or interactive capability may be used in accomplishing these analyses. The System may be used in various ways to support its primary purpose. Supporting uses of the System include maintenance of the Master Data Base, assessment of the cost of an analysis run, and maintenance of the Master Command File.

#### 4.2 MASTER DATA BASE

The Master Data Base contains descriptions of aircraft components and other analysis components to be analyzed; descriptions of coupling between and among components; descriptions of maneuvers, conditions, and operating regimes for the analysis; and descriptions of failure/damage effects. Information is created or changed in the Master Data Base only by an authorized person through use of the Data Base Support Package of the Support Complex. Information is read from the Master Data Base during the input phase of an analysis run to create the Run Data Base. The System user specifies in user input the data in the



Master Data Base that are to be used in the analysis run. The System user also has the capability to change the data read from the Master Data Base for the duration of the analysis run. The Run Data Base created for an analysis run contains the data selected from the Master Data Base by statements in user input. Section 4.3 describes the user input statements that cause selection and modification of data from the Master Data Base.

The data in the Master Data Base are organized into a hierarchy of sets of data. A set is a collection of data that is identified and managed as a single entity for convenience and efficiency. A set may contain any combination of three kinds of elements: other sets, arrays of data-items, and individual data-items. A set contained within a set has characteristics identical to the containing set. (To avoid the awkward nomenclature of "contained set" and "containing set" in the text that follows, a set contained in another set is referred to as a "subset." This does not identify a new type of element in the Master Data Base.) An array of data-items is a collection of data-items that are identified by an array name and an index in the array rather than by individual names. The indexes may be multi-dimensional. A data-item is the smallest identifiable element in the Master Data Base. Data-items that are not members of arrays are identified by names.

Each set, array, or data-item in the Master Data Base has a name. Sets in the Master Data Base are named at the time the sets are created. The conventions that are used in naming sets are controlled at each installation and may be as rigid or as flexible as the installation chooses. Sets will usually be named in a way to remind the user of the object or condition described by the set. For example, a set containing the description of a main rotor for an aircraft with model number UH60A might be named UH60AMR or a set describing a sinewave gust response maneuver might be named GUSTSIN20. The hierarchical structure of the Master Data Base is useful for representing a physical configuration that consists of components with subcomponents with sub-subcomponents to as fine a level of detail as

is necessary for a complete representation of an aircraft or other physical configuration. Using this strategy, an aircraft can be represented as a single set named, for example, UH60A. This set contains sets that describe components of the aircraft. For example, some of the subsets might be UH60AMR, mentioned above; UH60ACT, describing the control system; and UH60AAF, describing the airframe.

The arrays and individual data-items in the Master Data Base contain the values that represent the physical characteristics of the physical configuration or other entity being described. These values must be supplied to the software elements of the Technology Component in order for them to perform the required analysis. The array names and data-item names are the primary means of specifying the values that are to be made available to each individual software element. For this reason the names of arrays and data-items may not be arbitrarily assigned when a set is created. Every set describing a rotor, regardless of the name assigned to the rotor and, hence, to the set, must have arrays or data-items with specific names in order for the rotor to be simulated and analyzed by the System. Generally, the names will be in accord with the document, Nomenclature for the Second-Generation Comprehensive Helicopter Analysis System.<sup>36</sup>

Because the Data Base Management Subsystem is used to retrieve data from the Master Data Base, it is possible to rearrange the arrays and data-items within a set and even to add new arrays or data-items without disturbing the software elements performing the analysis. The necessary link between the System and the data is the name of the array or the data-item.

Just as each data element (set, array, or data-item) of the Master Data Base has a name, each data element also has a type. The type of a data element is

---

<sup>36</sup> NOMENCLATURE OF THE SECOND-GENERATION COMPREHENSIVE HELICOPTER ANALYSIS SYSTEM, Applied Technology Laboratory, U.S. Army Research and Technology Laboratories, Fort Eustis, Virginia, to be published.

assigned when the element is created. The type of a set is similar in meaning to the type of the entity described by the set. For example, an aircraft of type SRH (single-rotor helicopter) has certain components, including an engine and a control system. A set of type SRH, describing an aircraft of type SRH, has sets corresponding to the aircraft components, including a set of type ENGINE and a set of type CONTROL. The subdivision of types can be carried to the level of detail necessary for a complete description of the aircraft.

Within the Master Data Base, the type of a set specifies an entry in the dictionary that describes the characteristics of all sets of that type which are in the Master Data Base. Among the characteristics in the dictionary are the number and the type of all subsets and the number and the type of arrays and data-items in the set. Each name of a data element is also in the dictionary, along with the type of the named data element. It is through the use of the dictionary that the Data Base Management Subsystem retrieves the required sets, arrays, or data-items from the Master Data Base.

An example of some sets in a Master Data Base, along with their associated names and types, is given in Figure 35. In this example, six sets are defined at the highest level of the hierarchy. Four of the six sets describe aircraft of types SRH (single-rotor helicopter), TRH (tandem-rotor helicopter), and CRH (coaxial-rotor helicopter). The other two sets at the highest level of the hierarchy describe components of type ROTOR and AEROSURF (aerodynamic surface). The DMG at the end of the names of these last two sets is a reminder that the sets describe a damaged main rotor and aerodynamic surface, respectively, for helicopter model UH60A. Some of the subsets of set UH60A are shown in this example. Note that there are two sets of type ROTOR and that to distinguish between them the type has been qualified as ROTOR(MAIN) or ROTOR(TAIL). Set UH60AAF, which describes the airframe, is divided into subsets that represent components of the airframe. Those shown in the example represent components FUSE (fuselage), AEROSURF (aerodynamic surface), and



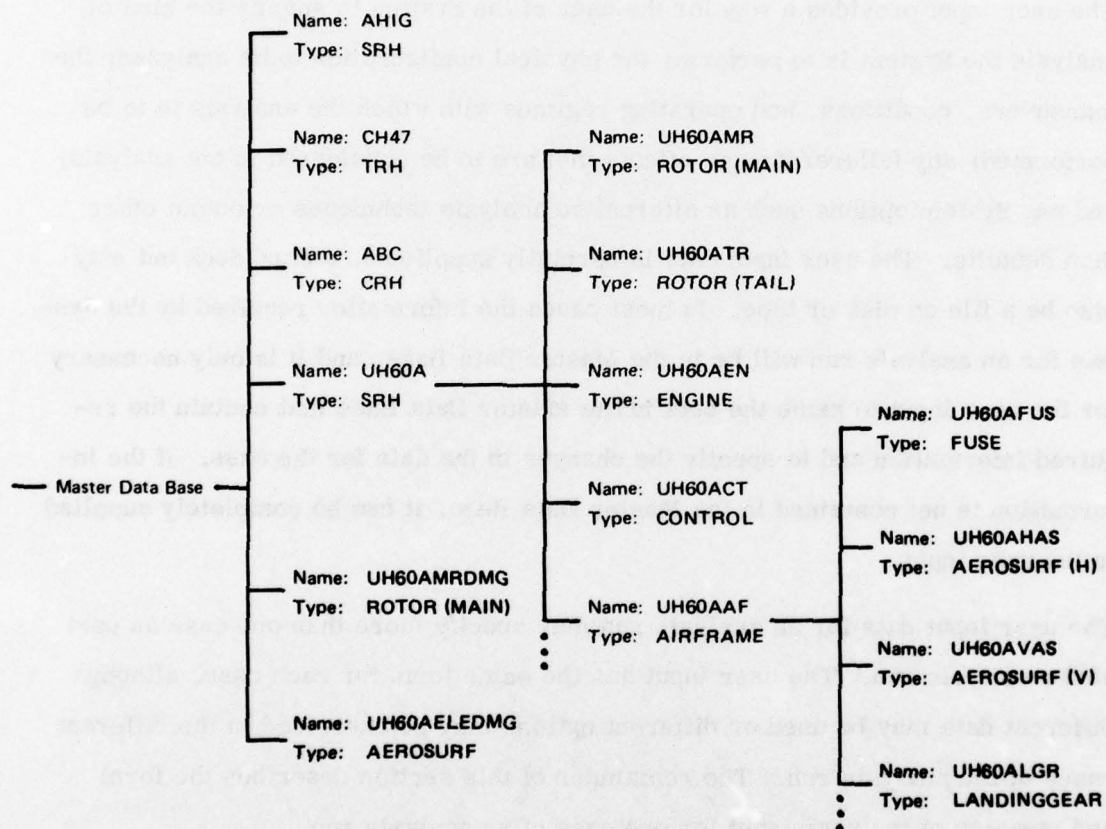


Figure 35. Example of Sets in the Master Data Base

LANDINGGEAR (landing gear). There are two sets of type AEROSURF that are distinguished by qualifier. In this example the set names are indicative of the data contained in the sets. The extent to which this practice is followed at an installation is completely under installation control.

#### 4.3 USER INPUT

The user input provides a way for the user of the System to specify the kind of analysis the System is to perform; the physical configuration to be analyzed; the maneuvers, conditions, and operating regimes with which the analysis is to be performed; any failure/damage effects that are to be considered in the analysis; and any System options such as alternative analysis techniques or output other than defaults. The user input data is normally supplied as a card deck but may also be a file on disk or tape. In most cases the information required by the System for an analysis run will be in the Master Data Base, and it is only necessary for the user input to name the sets in the Master Data Base that contain the required information and to specify the changes in the data for the case. If the information is not contained in the Master Data Base, it can be completely supplied in the user input.

The user input data for an analysis run may specify more than one case as part of the analysis run. The user input has the same form for each case, although different data may be used or different options may be exercised in the different cases of the analysis run. The remainder of this section describes the form and meaning of the user input for one case of an analysis run.

The user input data for a case is divided into six sections:

1. Case Specification Section
2. Configuration Specification Section
3. Conditions Specification Section
4. Failure/Damage Specification Section

5. Options Specification Section
6. Accuracy Assessment Specification Section

The Case Specification Section must be present in the user input for each case, and the Configuration Specification Section and Conditions Specification Section must be present in the first case of an analysis run. All other sections of user input are optional; each section is present only if information in that section is required for the analysis to be performed.

Each section of user input begins with a word indicative of the name of the section followed by a colon. The introductory words are

1. CASE:
2. CONFIGURATION:
3. CONDITIONS:
4. FAILURE DAMAGE:
5. OPTIONS:
6. ACCURACY:

Each of the six keywords that introduce a section must be the first word on the card or card image that is the first card of the section. With that exception, the user input data is free-form; that is, it can begin in any column, and multiple statements may appear on a card. This enables the System user to arrange the user input in an easily readable format. The six sections of user input data are specified in Sections 4.3.1 through 4.3.6.

#### 4.3.1 Case Specification Section

The Case Specification Section identifies the analysis run and the case and specifies the type of analysis to be performed for this case. The Case Specification Section has four kinds of statements. The first three statements are as follows:

CASE:

ANALYSIS RUN ID identification of analysis run.

CASE ID identification of this case.



The next statement specifies three items: aircraft life cycle phase and aircraft technical characteristic, cost assessment (if desired), and units to be used for input data.

An example of a full Case Specification Section is as follows:

CASE:

ANALYSIS RUN ID PERFORMANCE OF UH60A.

CASE ID ROTOR ZZ435.

DETAILED DESIGN PERFORMANCE, ASSESS COST, METRIC  
UNITS

The statements shown here are written on separate lines and indented. This form of preparation of the user input data is recommended for readability and ease of change, but it is not required by the System.

Under certain conditions, some of the statements of the Case Specification Section may be omitted. The first case of an analysis run must identify the run, but the ANALYSIS RUN ID statement may be omitted from all cases after the first. If there is only one case, the CASE ID statement may be omitted. The phrase ASSESS COST is included only if a cost estimate is desired for this case. The last phrase in the last statement may be either ENGLISH UNITS or METRIC UNITS. If English units are to be used, the phrase may be omitted.

The example given above identifies the analysis run as PERFORMANCE OF UH60A. The identification of the analysis run is printed as a primary heading on each page of printed or plotted output from the run. The case above is identified as ROTOR ZZ435. The case identification is printed as a secondary heading on each page of printed or plotted output from the run. Presumably other cases in this analysis run will analyze the performance of UH60A with other rotors. The last line of the example shown above specifies that the System is to make a PERFORMANCE analysis for the DETAILED DESIGN life

cycle phase. (A particular life cycle phase is indicative of a particular level of complexity in the analysis.) The last line further specifies that the System, instead of actually performing the analysis specified for this case, is to estimate the cost of performing the analysis and report the estimated cost to the user as the primary output for this case. The last phrase of the last line of the example above, METRIC UNITS, indicates that any values which appear in the following sections of user input data are in metric units.

Three additional examples of the Case Specification Section follow.

Example 1:

CASE:

CASE ID ROTOR XY570.  
RESEARCH LOADS AND VIBRATIONS

This Case Specification Section is valid only if it is not the first case of an analysis run, because the ANALYSIS RUN ID statement is missing. This example specifies that for case ROTOR XY570, the System is to perform a LOADS AND VIBRATIONS analysis for the RESEARCH life cycle phase. All values in user input are in English units because there is no indication to the contrary.

Example 2:

CASE:

ANALYSIS RUN ID PRELIMINARY VIBRATIONS.  
PRELIMINARY DESIGN LOADS AND VIBRATIONS, METRIC  
UNITS

This Case Specification Section is for an analysis run with only one case because there is no CASE ID statement. (Note that even though there is only one case it is permissible to include the CASE ID statement.) For this case the System is to perform a LOADS AND VIBRATIONS analysis for the

PRELIMINARY DESIGN life cycle phase. Values in user input data are in METRIC UNITS.

Example 3:

CASE: ANALYSIS RUN ID PROJECT ANALYZE--DAY43--RUN14.  
CASE ID AEROELELASTIC STABILITY. RESEARCH AEROELASTIC  
STABILITY

In this example the user has chosen to run all his statements together for his own reasons. This Case Specification Section causes the System to perform an AEROELASTIC STABILITY analysis for the RESEARCH life cycle phase. The values in user input data are in English units.

#### 4.3.2 Configuration Specification Section

The Configuration Specification Section describes the configuration to be analyzed. This section includes data for aircraft components, other analysis components, and coupling of components. The configuration to be analyzed may be a complete aircraft or a part of an aircraft. Examples of the latter are a rotor (possibly with major drive system components) mounted on a whirl stand, and a combination of components in a wind tunnel.

The introductory statement to the Configuration Specification Section must be the first word on a card and is as follows:

##### CONFIGURATION:

If the data describing the configuration to be analyzed are in the Master Data Base, the next statement in the Configuration Specification Section is a set specification statement that gives the type of configuration to be analyzed and the name of the set from the Master Data Base that contains the data for that configuration. The form of the set specification statement is as follows:

type = set name



The set specification statement may be on the same card as CONFIGURATION: or on a succeeding card; for example,

CONFIGURATION: SRH = UH60A

means that the type of physical configuration to be analyzed is a single rotor helicopter and that the set with name UH60A contains the data describing the configuration. The System will verify that set UH60A is of type SRH. If the System finds a type mismatch, it produces a diagnostic message informing the user of the discrepancy. For a discussion of set types, see Section 4.2. For another example,

CONFIGURATION:

MRWR = RWS53

means that the configuration to be analyzed is of type MRWR (main rotor on a whirl rig) and that the set with the name RWS53 describes the configuration.

The type of the configuration named in the set specification statement must be one of a list of types of configuration that the System can analyze. The types of configuration that the System can analyze and, hence, the types of sets that can be named in the set specification statement are equivalent to the Detailed Functional Capability Physical Systems defined in the Government Draft on Detailed Functional Capabilities (i. e., tables from Section 30 of the Type A System Specification). For example, sets describing an aircraft-only physical system are of types SRH (single-rotor helicopter), TRH (tandem-rotor helicopter), CRH (coaxial-rotor helicopter), etc. Sets describing a main rotor on a whirl rig are of type MRWR.

Once a set with a valid type is named in the set specification statement of the Configuration Specification Section, a wealth of information is available to the System about the configuration to be analyzed. The entire hierarchy of data beginning with the named set and ending, after a variable number of hierarchical levels of subsets, in data-items or arrays of data-items conveys information about the configuration to be analyzed and about the software elements that will

be called to perform the analysis. If the configuration is a rotor on a whirl stand, then there is no need to call on software elements that simulate landing gear; or if the configuration is a multiple-rotor helicopter, then the software element that models the main rotor must be executed a number of times equal to the number of rotors. The specification of this set name will automatically cause the proper software elements to be executed.

The next statements in the Configuration Specification Section specify the replacement of any of the subsets of the aforementioned set with another set of the same type in the Master Data Base. To accomplish this replacement, a set specification statement of the same form as described above is used, separated from a previous set specification statement by a comma. An example of set replacement follows:

CONFIGURATION:

SRH = UH60A,

ROTOR = ZZ435

This example causes the single-rotor helicopter described in UH60A to be analyzed with its rotor replaced by ZZ435. The System verifies that the set of type SRH contains a subset of type ROTOR and that the new set, ZZ435, is a set of type ROTOR. Any discrepancy in set types causes the System to produce a user input diagnostic message.

For some types of configuration, there is more than one subset of a given type. For example, a configuration of type SRH (single rotor helicopter) has two subsets of type ROTOR (one for the main rotor and one for the tail rotor). This situation might make it impossible for the System to identify the subset to be replaced in the preceding example. To prevent the ambiguity, any subsets of a configuration that have identical types must have the types qualified to make the subsets identifiable by qualified type. In the example of the configuration of type SRH, the two subsets of type ROTOR are distinguished by qualifying

their types as ROTOR(MAIN) and ROTOR(TAIL). The foregoing example should have been written:

CONFIGURATION:

SRH = UH60A,

ROTOR(MAIN) = ZZ435

The example now specifies that the subset of type ROTOR(MAIN) is to be replaced by set ZZ435, which must have type ROTOR.

The following are some examples of the Configuration Specification Section as described to this point:

Example 1:

CONFIGURATION:

SRH = UH60A

, ROTOR(MAIN) = ZZ435

This example is the same as that described previously. The configuration to be analyzed is contained in set UH60A, which must be of type SRH. The only change to the data as contained in set UH60A in the Master Data Base is that the data for the main rotor is replaced by the rotor data contained in set ZZ435 in the Master Data Base, which must be of type ROTOR. The replacement statement for the rotor and the comma separating this statement from the previous statement are on a separate card so that by removing that one card the configuration can be changed to that contained in set UH60A with no rotor replacement.

Example 2:

CONFIGURATION:

MRWR = ROTEST23

In this example, the set named ROTEST23, which is of type MRWR (main rotor on whirl rig) is the configuration to be analyzed.



**Example 3:**

**CONFIGURATION:**

TRH = XPER70,  
ROTOR(FORWARD) = XRTR15,  
ROTOR(AFT) = XRTR15,  
ROTORCOUP = ROTOR-TO-ROTOR-INTERFERENCE-15

In this example, the configuration to be analyzed is of type TRH (tandem rotor helicopter). The set describing the configuration is named XPER70. Both rotors are replaced by set XRTR15, which is of type ROTOR, and the set named ROTOR-TO-ROTOR-INTERFERENCE-15, which is of type ROTORCOUP, contains the data for rotor-to-rotor coupling.

Thus far, the examples provided have assumed that a single name is enough to uniquely identify a set in the Master Data Base. In fact, in order to identify a set in the Master Data Base, it is necessary to name the sets in the hierarchy containing the set to be identified; for example,

UH60A.UH60AEN

This example identifies set UH60AEN as a subset of set UH60A. Any of the set names in the examples given above must appear in this form unless they are at the highest level of the hierarchy. Thus, example 3 (above) might appear as

**CONFIGURATION:**

TRH = XPER70,  
ROTOR(FORWARD) = XPER53.XRTR15,  
ROTOR(AFT) = XPER53.XRTR15,  
ROTORCOUP = ROTOR-TO-ROTOR-INTERFERENCE-15

In this example, set XRTR15 is a subset of set XPER53.

The same type of ambiguity might occur in specifying the type of a set to be replaced if, for example, a blade of a rotor of a tandem-rotor helicopter configuration is to be replaced. In this example, there are two rotors and multiple blades per rotor so that to specify which blade of which rotor is to be replaced requires the following:

ROTOR(FORWARD).BLADE(C) = XBLAD5

In this example, XBLAD5 is a set at the highest level of the hierarchy in the Master Data Base that is to be used as a replacement for a blade of the forward rotor in this configuration.

To this point, the discussion has been limited to sets and subsets. We now turn our attention to the arrays and data-items of which sets are composed. The statements described below are called value specification statements. They enable the System user to set the values of data-items or arrays for this case. A value specification statement has the following form: data-item identification, equals sign, value. Just as for set names and types, it is possible for duplicate data-item names to occur in different sets. For example, angle of attack for different aerodynamic surfaces is designated ALPHA in the sets describing the different aerodynamic surfaces. To identify the data-item, it is necessary to precede its name with the types of the sets containing the data-item. For example,

SRH.ROTOR(TAIL).XTR = 12.48

means that the X coordinate of the location of the tail rotor is 12.48 meters for this case (assuming METRIC UNITS were specified in the Case Specification Section).

For another example,

TRA.ASURF(G).ALPHAG = 0.05

indicates that the geometric angle of attack for aerodynamic surface G of a tilt-rotor aircraft is 0.05 radian for this case.

A data-item contained in an array is identified by the array name followed by indexes in parentheses. For example,

SRH. ROTOR(MAIN). STRUCPROP. EIC(10) = 2.23E8

means that the tenth data-item of the chordwise bending stiffness array, which is in the indicated set hierarchy, is to be set to  $2.23 \times 10^8$  pound-inches<sup>2</sup>.

In another example,

SRH. AIRFRAME. FUSE. OMEGAF(3) = 13.7

means that the third fuselage natural frequency is to be set to 13.7 hertz.

It is in the value specification statements that the question of units becomes important. The units specified in the Case Specification Section are assumed for all values unless otherwise specified in the value specification statement. The value on the right of the equals sign may be followed by a units designation to indicate the units of this particular value. For example, the geometric angle of attack specified above might instead be given in degrees, as follows:

TRA. ASURFG. ALPHAG = 3 DEG

It is possible to specify an entire configuration in user input without reference to the Master Data Base by omitting the set specification statements from the Configuration Specification Section and using only value specification statements to set all the initial values required for the analysis. If it is necessary to specify large configurations frequently without reference to the Master Data Base, the use of the data-item identification on the card in addition to the value adds bulk to the user input deck. This approach has been taken here in spite of that disadvantage because of its overriding advantages. Because each value is separately identified, the order of cards is unimportant provided they are in the correct section of user input. The identification of each value further adds documentation to the user input deck, which reduces errors in its creation



and in any changes required later. In the final design of the user input, abbreviations will be defined that will alleviate the bulk problem by allowing shorter ways to specify values (i.e., by setting an entire array in a single statement). These measures will increase the advantages of the approach proposed here.

In summary, the physical configuration to be analyzed is specified in the Configuration Specification Section of user input. The first statement of the Configuration Specification Section is the word CONFIGURATION followed by a colon. The next statements of the section are set specification statements if the configuration is in the Master Data Base. The final statements of the section are value specification statements if any values in the configuration are to be specified in user input. The sets specified in set specification statements are moved from the Master Data Base to the Run Data Base, and all values named in value specification statements are set to the values given.

#### 4.3.3 Conditions Specification Section

The Conditions Specification Section describes the maneuvers, conditions, and operating regimes to be applied to the configuration for the analysis. The first statement of the Conditions Specifications Section is as follows:

##### CONDITIONS:

The remainder of the Conditions Specification Section is identical in form to the Configuration Specification Section described above; that is, sets in the Master Data Base describing the maneuvers, conditions, and operating regimes are selected for inclusion in the Run Data Base using the same form of set specification statement as described above; and values in those sets are changed by the use of value specification statements, which have the same form as the value specification statement described above.

As a minimum, the conditions for an analysis must be specified. If no maneuvers are specified, a steady-state operation is assumed. If maneuvers are

specified, System Commands that name appropriate software elements from the Maneuver Subsystem will be included in the Sequence Control Table.

It is possible to perform analyses with no more in user input than the Case Specification Section, the Configuration Specification Section, and the Conditions Specification Section. The User Input Package, which reads and processes user input, uses the information in the Configuration Specification Section and the Conditions Specification Section to construct a Run Data Base (described in Section 4.6). The sets specified in set specification statements are moved from the Master Data Base to the Run Data Base, and all values named in value specification statements are set to the values given. The configuration thus specified, together with information from the Case Specification Section as to aircraft life cycle phase and aircraft technical characteristic, indicates which subsequences of System Commands (see Section 2.5) are to be selected from the Master Command File and concatenated to form the sequence of System Commands in the Sequence Control Table. Section 4.5 describes System Commands.

#### **4.3.4 Failure/Damage Specification Section**

The Failure/Damage Specification Section specifies failure or damage effects that are to apply to the configuration for this case. If no failure or damage effects are applicable for this case, then the Failure/Damage Specification Section will not appear in user input. The first statement of the Failure/Damage Specification Section is as follows:

##### **FAILEDAMAGE:**

The remainder of the Failure/Damage Specification Section is identical in form to the Configuration Specification Section described above; that is, sets in the Master Data Base describing the failure or damage effects for this case are

selected for inclusion in the Run Data Base using the same form of set specification statement, and values of data elements in those sets are changed by the use of value specification statements.

The result of failure/damage effects on the creation of the sequence of System Commands in the Sequence Control Table is to create a separate subsequence of System Commands. This separate subsequence is given control at the point in the analysis when the failure or damage occurs. The flow of control is generally unchanged, but the flow of data as specified in the System Command subsequence will change as a result of the failure or damage.

#### **4.3.5 Options Specification Section**

The Options Specification Section describes the options the user wishes to exercise for this case. The Options Specification Section may be omitted if the default options are appropriate for this run. The first statement of the Options Specification Section is as follows:

##### **OPTIONS:**

The Options Specification Section is divided into the following subsections, each of which specifies a different type of option:

1. Print Specification Subsection
2. Plot Specification Subsection
3. Analysis Technique Specification Subsection
4. Diagnostic Specification Subsection
5. Checkpoint/Restart Subsection

The Print Specification Subsection specifies quantities to be printed in addition to those printed by default and the format in which they are to be printed. In addition, this subsection specifies quantities normally printed by default that



the user chooses not to be printed for this case. The forms of the statements for this subsection are as follows:

PRINT REPORT a

PRINT INTERMEDIATE IN FORMAT b:

data element, ... , data element

PRINT FINAL IN FORMAT c:

data element, ... , data element

DO NOT PRINT:

data element, ... , data element

FORMAT d:

format definition statement

The PRINT statements specify that certain data elements are to be printed that would otherwise not be printed. The word REPORT followed by an identification refers to a collection of data elements that are normally printed together in an internally defined format. The word INTERMEDIATE means that the specified data elements are to be printed whenever they are produced by a software element during the analysis. The word FINAL means that the specified data elements are to be printed at the end of the analysis for this case. If neither word appears, FINAL is assumed. Several print formats are internally defined in the System. If these formats are used, the FORMAT statement may be omitted. A print format not internally defined in the System may be defined in the use of the FORMAT statement. Examples of the use of these statements are as follows:

PRINT IN FORMAT 4:

LOADS.AUTOCORR.AIRFRAME.NODE(5)

means print the autocorrelation of the vibration of node point 5 on the airframe in format 4 (an internally defined format) at the end of processing for this case.

DO NOT PRINT:

AEROSTAB. EIGENVECTORS. INERTIAFORCES

AEROSTAB. EIGENVECTORS. AEROFORCES

means suppress the printing of the inertia forces and aerodynamic forces calculated as the result of the analysis.

The Plot Specification Subsection specifies quantities to be plotted and the format in which they are to be plotted. The forms of the statements for this subsection are as follows:

PLOT INTERMEDIATE IN FORMAT x:

array, ... , array

PLOT FINAL IN FORMAT y:

array, ... , array

FORMAT z:

format definition statement

The two PLOT statements have meanings similar to the two PRINT statements in the Print Specification Subsection. Just as for print formats, the System has several internally defined plot formats for which no FORMAT statement is necessary. Other formats can be defined in user input by means of the Format Definition Statement defined by the user. The effect of the Plot Specification Subsection is to produce device-independent plot files. The device-independent plot files can be processed during the analysis to produce a plot for a specific device or they can be subsequently processed to produce plots on off-line devices.

The Analysis Technique Specification Subsection specifies which of several alternative analysis techniques the System is to use in performing the analysis. The statements in this subsection are a list of keywords. The appearance of

a keyword specifies that a particular analysis technique is to be used. The keywords that are currently defined and their meanings are as follows:

<u>Keyword</u>	<u>Meaning</u>
MODAL	Use the modal method for the solution of dynamic equations
DIRECT	Use the direct method for the solution of dynamic equations
TRANSFER	Use the transfer matrix method for the solution of dynamic equations
UNCOUPLED	Iterate to trim using uncoupled equations
FLYTOTRIM	Use an autopilot to fly the aircraft to trim
FLOQUET	Use Floquet theory to solve for aeroelastic stability
LINSTAB	Use constant coefficient equations to solve for aeroelastic stability
THEODORSEN	Use the Theodorsen method for unsteady aerodynamic calculations
CARTA	Use the Carta method for unsteady aerodynamic calculations

This subsection, along with the Case Specification Section and the Configuration Specification Section, affects the construction of the Sequence Control Table. The analysis technique to be used is another determining factor used by the User Input Package to select subsequences of System Commands from the Master Command File and construct the sequence of System Commands in the System Control Table.

The Diagnostic Specification Subsection specifies the options to be used for producing diagnostic messages for this case. The options are expressed as



short statements which are for the most part self-explanatory. The following statements may be used:

SUPPRESS INFORMATIVE DIAGNOSTICS

SUPPRESS WARNING DIAGNOSTICS

SUPPRESS DIAGNOSTICS

STOP IF ERROR IN INPUT

WRITE DIAGNOSTICS ON FILE a

The first two statements are self-explanatory. The third statement is a shorter way to achieve the effect of both the first and second statements. There is no option to suppress fatal diagnostic messages. The fourth statement causes processing to halt after checking the input if any errors are discovered in the input. (This option affects only "warning" errors. Fatal errors always stop processing.) The last statement causes the diagnostic messages to be written on a file which may be examined after the analysis run.

An additional capability that may be exercised in the Diagnostic Specification Subsection is to plot graphable input. The statement that uses this function is similar to the PLOT statements discussed in the Plot Specification Subsection:

PLOT INPUT IN FORMAT x:

array, array, ... , array

FORMAT y:

format definition statement

As before, some plot formats are internally defined in the System, but if a different format is desired, it may be defined in user input.

The Checkpoint/Restart Subsection specifies the use of checkpoint or restart functions for this case. The statements which may be used in this section are as follows:

CHECKPOINT ON FILE b

RESTART FROM CHECKPOINT n ON FILE c

The occurrence of the CHECKPOINT statement causes the checkpoint System Commands in the Sequence Control Table to be executed for this case. Each time a checkpoint is taken, it is given an identification by the System. The checkpoint identification is included in user output and is used in the RESTART statement as shown above if restart from that checkpoint is desired in a subsequent run.

The occurrence of the RESTART statement causes the Run Data Base and the Sequence Control Table to be established from the specified checkpoint. Changes are then made to the Run Data Base and Sequence Control Table based on user input.

#### 4.3.6 Accuracy Assessment Specification Section

The Accuracy Assessment Specification Section specifies the output data elements to be assessed for accuracy, the input variables to be varied, and the range of variation. This section also contains (a) data against which System results are to be compared and (b) the error in that data.

The occurrence of the Accuracy Assessment Specification Section in the user input causes the User Input Package to create a sequence of System Commands for performing accuracy assessment.

#### 4.4 MASTER COMMAND FILE

The Master Command File contains subsequences of System Commands for use by the User Input Package in creating the sequence of System Commands to control a particular analysis run. System Commands are described in Section 4.5. The

Master Command File is changed only by an authorized person using the Data Base Support Package of the Support Complex.

The Sequence Control Table is a table internal to the System that contains the sequence of System Commands that specifies the actions to be taken by the System in performing an analysis. The User Input Package creates the Sequence Control Table from subsequences of System Commands in the Master Command File based on information in user input.

The overall sequence of System Commands to be included in the Sequence Control Table is determined by the combination of aircraft life cycle phase, aircraft technical characteristic, the existence (or nonexistence) of maneuvers, and the existence (or nonexistence) of failure/damage effects in user input. The selection of small groups of System Commands to be included in the overall sequence is determined by the configuration to be analyzed (from the Master Data Base as selected and modified by user input) and by the analysis technique and other options selected by user input. Examples of System Command Sequences were shown in graphical form in Section 2.5.

The Sequence Control Package of the Executive Component reads the System Commands from the Sequence Control Table during the processing phase and calls the proper software element to perform the action specified by each System Command.

#### 4.5 SYSTEM COMMANDS

Sequences of System Commands appear in the Master Command File and in the Sequence Control Table. Sequences of System Commands control the execution and data flow during the processing phase of an analysis run. The reason for using System Commands to direct the System through the steps required to perform an analysis is to allow the System the flexibility to perform many different analyses by changing only the sequence of System Commands. The sequence of System Commands controlling an analysis is normally created and



put in the Sequence Control Table by the User Input Package based on a description of the analysis contained in user input, as described in Section 4.3. The System can also accept user input directly in the form of System Commands as described below. The user of this form of input (a methods developer is a likely user of this form of input) is required to know more about the internal design of the System than is the user of the more conventional user input described in Section 4.3; furthermore, there are fewer internal System checks for errors in such input.

System Commands are of two types: execution and sequence control.

An execution command contains the name of the software element to be executed and the input data elements and the output data elements to be used by the software element for this execution. Upon encountering an execution command, the Sequence Control Package (which is responsible for the order of execution of System Commands) calls the Run-Time Control Package for execution of the software element named in the execution command. The Run-Time Control Package brings the named software element into memory (if it is not present), makes the input data elements available from the Run Data Base, and transfers control to the named software element. Upon completion of execution, the named software element returns control to the Run-Time Control Package, which stores the output data elements in the Run Data Base.

A sequence control command provides the capability to control the sequence in which software elements are invoked in the System Command Sequence. There are three forms of sequence control commands:

1. IF-THEN-ELSE
2. DO-WHILE
3. STOP

The IF-THEN-ELSE form of sequence control command has three separate System Commands that together perform its function:

1. IF Command
2. ELSE Command
3. ENDIF Command

The IF Command contains a condition that has a value of TRUE or FALSE. The condition is a relationship between variables or expressions. If the value is TRUE, the sequence of command execution continues with the next System Command. If the value is FALSE, the sequence of command execution continues with the ELSE Command or, if the ELSE Command is omitted, with the ENDIF Command.

The ELSE Command appears optionally after an IF Command. The value of the condition that was part of the IF Command determines the action to be taken by the ELSE Command. If the value is TRUE, the System Commands between the IF Command and the ELSE Command have been executed, and the effect of the ELSE Command is to cause the sequence of command execution to continue from the ENDIF Command. If the value is FALSE, the ELSE Command causes the sequence of command execution to continue at the System Command immediately following the ELSE Command.

The ENDIF Command must always follow an IF Command regardless of whether or not the ELSE Command is used. The purpose of the ENDIF Command is to mark the place in the System Command Sequence at which command execution is to continue regardless of the effects of the IF Command or the ELSE Command. The effect of the ENDIF Command is to cause the sequence of command execution to continue with the next System Command.

The DO-WHILE form of Sequence Control Command has two separate System Commands that together perform its function:

1. WHILE Command
2. ENDDO Command

The WHILE Command contains a condition identical in form to the condition in the IF Command. This condition can have a value of TRUE or FALSE. If the value is TRUE, the sequence of command execution continues with the next System Command. If the value is FALSE, the sequence of command execution continues with the System Command following the ENDDO Command.

The ENDDO Command must always follow a WHILE Command. The ENDDO Command causes the sequence of command execution to continue with the associated WHILE Command.

The STOP Command causes System Command execution to stop. Control is transferred either to the User Output Package of the User Interface Subsystem for printing or plotting the results of the analysis or to the user at an interactive terminal.

#### 4.6 RUN DATA BASE

At the end of the input phase, the Run Data Base contains the initial descriptions of aircraft components; other analysis components; coupling of components; maneuvers, conditions, and operating regimes; and failure/damage effects required as input for a single analysis run. During the processing phase, the Run Data Base contains intermediate results produced by software elements during an analysis run.

The organization of sets of data that exist at the beginning of the processing phase of an analysis run is the same as their organization in the Master Data Base. For any particular analysis run, only those sets required for the analysis are contained in the Run Data Base.



The User Input Package creates the input sets in the Run Data Base and sets their values from the Master Data Base and from user input. The User Input Package also creates the sequence of System Commands in the Sequence Control Table from commands in the Master Command File based on information in user input and the Master Data Base. The sequence of System Commands to be executed specifies the data required by, and produced by, the software elements to be executed. From this information the User Input Package defines the sets in the Run Data Base that will hold intermediate results for this run. The definition of all sets to be used in this analysis run, including the location of each set, is contained in a dictionary for the Run Data Base. The Run Data Base dictionary is created by the User Input Package and is used by software elements of the Data Base Management Subsystem to store or retrieve data throughout the analysis run.

#### 4.7 OUTPUT DATA

Output data are data which are printed or plotted or displayed on a terminal for interpretation by the user. Output data also include data stored in files on disk or tape by the System for potential later use. Data for external models, checkpoint data, and optionally, diagnostic messages are included in this category.

Output data are produced by the User Output Package, by the Checkpoint Package, or by a software element in the External Models Interface Subsystem. Any of these software elements is called in the usual way, i. e., by being named in a System Command and by being passed input data. These software elements differ from the typical software element in that they write information onto files using the File Management Package of the Operating System Service Subsystem.

Printed user output is formatted by the System into reports that are compact and easy to read. The format of the printed output from an analysis run depends on the quantities that are output from that run, which depends on the options specified in user input. The units for the printed output are the same as the units of the input quantities in the Case Specification Section, unless specified otherwise in the Options Specification Section. Regardless of the format and quantity of the output variables, each page of printed output has header information that contains the title of this analysis run, the title of the case, the date of the run, the page number, the report title, and column headings if the output is arranged in columns.

User output plots may be in the form of X-Y plots, multivariable plots, contour maps, or three-dimensional projections, depending on the variables to be plotted and on user requirements. Regardless of the form of the plot or the device on which it is made, all plots contain the titles of the analysis run and case that produced them and the date. The variables plotted and identified, and the scale of the plot is shown.

Checkpoint data are stored on the Checkpoint File by the Checkpoint Package. The sets, arrays, or data-items in the Run Data Base that have been changed since the last call on the Checkpoint Package are written on the Checkpoint File. Upon restart, the Run Data Base can be restored from the Checkpoint File to its state at any checkpoint up to the last one executed so that the packages and subpackages in the previous analysis run that were required to obtain the state of the Run Data Base need not be executed.

#### 4.8 DIAGNOSTIC MESSAGES

Diagnostic messages from the System are of three types:

1.    Graphs of input data
2.    User input diagnostic messages
3.    Internal System diagnostic messages

Graphs of input data are produced, at user option, from data in the Run Data Base at the end of the input phase. These graphs are produced in the same manner and are identical in form to the user output plots discussed in Section 4.7. This option is exercised by a statement in the Option Specification Section of user input, described in Section 4.3.5.

User input diagnostic messages report to the user any error in user input discovered by the System. The user input diagnostic messages also report to the user unusual conditions or other information that may be useful. Most errors in user input will be discovered in the input phase, but some errors can be discovered only in the processing phase. User input diagnostic messages are of three types:

1. Informative--reports an unusual condition or other useful information
2. Warning--reports an error in input that may not be serious enough to prevent System execution
3. Fatal--reports an error that prevents System execution

The form of a user input diagnostic message is a clear, English-language statement of the error or condition detected in terms related to the user input. Internal System terminology such as module names will not appear in user input diagnostic messages.

Internal System diagnostic messages report internal logic errors in the System. Included as part of the message is any information collected thus far by the System that will help in locating and correcting the System error. The form of an internal System diagnostic message is an English-language statement identifying the message as an internal System diagnostic message followed by information intended to be used in the error correction process. Internal System diagnostic messages may be informative, warning, or fatal, just as are user input diagnostic messages.



## SECTION 5 - DEVELOPMENT PLAN

The development plan overview presented in this section summarizes CSC's current plan for the development of the First Level Release, Second Level Release, and Long Range System capabilities of the Second Generation Comprehensive Helicopter Analysis System. This plan is based on the Baseline Development Plan for the Second Generation Comprehensive Helicopter Analysis System, CSC/SD-78/6008, hereinafter called the Baseline Development Plan. Because of the need to effectively control the development of the System, CSC recommends that the Government require the Development Phase Contractor to deliver, as a Contract Data Requirements List (CDRL) item 2 months after Development Phase contract award, a System Development Plan. The System Development Plan would be an evolutionary conclusion to the planning activities of the Predesign Phase and to the source selection process that culminates with the award of the Development Phase contract to the Development Phase Contractor. CSC envisions that the System Development Plan evolves from the Baseline Development Plan; defines the responsibilities of the various organizations involved in System development; and is maintained as a baseline document so that up-to-date information about all technical and management aspects of the System's development are available to all concerned Government and industry organizations.

This development plan addresses the 19 items presented in Task I(e) of the Statement of Work of the contract.

Section 5.1, Organization and Responsibilities, defines the organizational structure and the responsibilities of the Applied Technology Laboratory, the prime contractor software firm (called the Development Phase contractor), the helicopter firm team member (called the integrated team member subcontractor), the CPCI subcontractors, the Government/Industry User Community, the Government/Industry Working Group, and Technical Advisory Group. This section is based on the Organization Plan of the Baseline Development Plan.

Section 5.2, Management Plans, is made up of the following subplans: Development Control Plan, Work Management Plan, Communication Plan, Internal Review and Reporting Plan, Configuration Management Plan, and Documentation Management Plan. These subplans are based on similarly titled subplans in the Baseline Development Plan.

Section 5.3, Technical Plans, is made up of three major subplans (Quality Assurance Plan, Testing Plan, and Documentation Plan) and five subordinate subplans (Data Processing Facility Plan, Data Management Plan, System Installation and Release Plan, Training Plan, and Maintenance Plan). These subplans are based on similarly titled subplans in the Baseline Development Plan.

Section 5.4, Implementation Plan, presents a time-phased plan for developing the First Level Release and the Second Level Release of the System.

Table 3 shows the relationship of the 19 items of Task I(e) of the Statement of Work to the subplans listed above.

## 5.1 ORGANIZATION AND RESPONSIBILITIES

The multiorganization (Applied Technology Laboratory, Development Phase Contractor, integrated team member subcontractor, CPCI subcontractors, Government/Industry User Community, Government/Industry Working Group, Technical Advisory Group) environment of the Development Phase makes it necessary to define and adhere to clear lines of authority and responsibility and effective lines of communication. The nature of this environment and a preliminary plan for defining these interfaces are presented in Section 5.1.1. The preliminary Project organization for the Development Phase is presented in Section 5.1.2 where the responsibilities of the Development Phase Contractor, the integrated team member subcontractor, and CPCI subcontractors are presented in relationship to the technical activities associated with each of the elements of the planned Project organization.

Table 3. Allocation of the 19 Items of Paragraph (e) of Task I of the Statement of Work to the Subplans of the Development Plan (1 of 3)

ITEM NUMBER	DESCRIPTION	SUBPLAN NAME AND SECTION NUMBER
1	ORGANIZATIONAL RESPONSIBILITY AND STRUCTURE OF THE CONTRACTOR AND SUBCONTRACTORS TEAM DESIGNING, PRODUCING, AND TESTING EACH CPCI	ORGANIZATION AND RESPONSIBILITIES, SECTION 5.1
2	MANAGEMENT AND TECHNICAL CONTROLS TO BE UTILIZED IN SOFTWARE DEVELOPMENT	MANAGEMENT PLANS, ESPECIALLY COMMUNICATION PLAN, INTERNAL REVIEW AND REPORTING PLAN, AND THE CONFIGURATION MANAGEMENT PLAN, SECTION 5.2; TECHNICAL PLANS, SECTION 5.3
3	METHODOLOGY FOR ENSURING SATISFACTORY DESIGN, TESTING, AND QUALITY ASSURANCE	QUALITY ASSURANCE PLAN, SECTION 5.3.1; AND TESTING PLAN, SECTION 5.3.2
4	APPROACH TO DEVELOP AND QUALIFY EACH CPCI	IMPLEMENTATION PLAN, SECTION 5.4
5	LEVEL OF MANPOWER ALLOCATED TO CONTRACTOR, SUBCONTRACTOR WORKING AS INTEGRATED TEAM MEMBER, CPCI SUBCONTRACTOR, AND GOVERNMENT-FUNDED CPCI DEVELOPER	IMPLEMENTATION PLAN, SECTION 5.4
6	SCHEDULE ALLOCATED FOR DEVELOPMENT OF EACH CPCI	IMPLEMENTATION PLAN, SECTION 5.4



Table 3. Allocation of the 19 Items of Paragraph (e) of Task I of the Statement of Work to the Subplans of the Development Plan (2 of 3)

ITEM NUMBER	DESCRIPTION	SUBPLAN NAME AND SECTION NUMBER
7	LEVEL OF MANPOWER ALLOCATED TO EACH TASK FOR EACH CPCI	IMPLEMENTATION PLAN, SECTION 5.4
8	IDENTIFICATION OF THE COMPUTER FACILITIES NEEDED TO SUPPORT DEVELOPMENT OF THE SYSTEM	DATA PROCESSING FACILITY PLAN, SECTION 5.3.4.1
9	APPROACH FOR CONTRACTOR AND SUBCONTRACTOR TESTING, INCLUDING RESOLUTION OF ERRORS, ETC.	TESTING PLAN, SECTION 5.3.2; AND MAINTENANCE PLAN, SECTION 5.3.4.5
10	TECHNIQUES FOR CONTROLLING DESIGN CONSISTENT WITH INTENT OF SYSTEM	QUALITY ASSURANCE PLAN, SECTION 5.3.1
11	METHODS AND PROCEDURES FOR COLLECTING, ANALYZING, MONITORING, AND REPORTING PERFORMANCE OF TIME-CRITICAL CPCIs	TESTING PLAN, SECTION 5.3.2
12	CONFIGURATION CONTROL OF MASTER TAPES, CARDS, AND DOCUMENTATION	CONFIGURATION MANAGEMENT PLAN, SECTION 5.2.5
13	APPROACH FOR MAINTAINING PROGRAMS DURING USER SYSTEM TESTING DURING VALIDATION PHASE	MAINTENANCE PLAN, SECTION 5.3.4.5

Table 3. Allocation of the 19 Items of Paragraph (e) of Task I of the Statement of Work to the Subplans of the Development Plan (3 of 3)

ITEM NUMBER	DESCRIPTION	SUBPLAN NAME AND SECTION NUMBER
14	APPROACH FOR DEFINING AND MANAGING THE DATA BASE TO BE USED FOR CPCI AND SYSTEM TESTING	TESTING PLAN, SECTION 5.3.2; DATA MANAGEMENT PLAN, SECTION 5.3.4.2
15	IDENTIFICATION OF SPECIAL SIMULATION, DATA REDUCTION, OR UTILITY TOOLS FOR USE IN DEVELOPMENT	IMPLEMENTATION PLAN, SECTION 5.4
16	APPROACH FOR ENSURING COMPUTER PROGRAM GROWTH, MODULARITY, AND EASE OF MAINTENANCE	QUALITY ASSURANCE PLAN, SECTION 5.3.1
17	APPROACH FOR DEVELOPING COMPUTER PROGRAM DOCUMENTATION	DOCUMENTATION PLAN, SECTION 5.3.3; AND DOCUMENTATION MANAGEMENT PLAN, SECTION 5.3.4.2
18	DATA PROCESSING FACILITY TO BE USED DURING DESIGN AND DEVELOPMENT	DATA PROCESSING FACILITY PLAN, SECTION 5.3.4.1
19	TRAINING APPROACH FOR FIRST LEVEL SYSTEM USERS	SYSTEM INSTALLATION RELEASE PLAN, SECTION 5.3.4.3; TRAINING PLAN, SECTION 5.3.4.4

#### **5.1.1 Development Phase Organizational Environment**

The Second Generation Comprehensive Helicopter Analysis System will be developed in a multiorganization environment, with numerous well-defined interfaces required.

The overall management of the development effort is the responsibility of the Applied Technology Laboratory.

The Development Phase contractor, expected to be one of the Predesign Phase contractors, is responsible for the following:

- Designing the System
- Identifying CPCIs
- Preparing a Type B5 Development Specification for each CPCI, for both First Level Release and Second Level Release capabilities
- Recommending those CPCIs to be developed by the Development Phase Contractor (in conjunction, possibly, with an integrated team member subcontractor), those by CPCI subcontractors, and those to be Government-furnished, based on the premises that few, if any, First Level Release CPCIs will be Government-furnished and few, if any, Second Level Release CPCIs will be developed by CPCI subcontractors
- Developing those CPCIs approved by the Contracting Officer
- Determining that each CPCI meets the requirements and quality assurance provisions of its Type B5 Development Specification
- Integrating all CPCIs into the System
- Conducting a functional demonstration of the System to demonstrate to Government and industry that the System meets the requirements and quality assurance provisions of the Type A System Specification



- Defining a unified documentation approach and editing documentation for each CPCI to promote the uniformly high standards needed for the engineering user and the methods developer
- Defining and implementing a configuration management plan
- Providing training and maintenance support to Government and industry users during the initial portion of the Validation Phase

The Government/Industry Working Group and the Technical Advisory Group continue their Predesign Phase activities in advising the Government project team.

The Government project team monitors the development of the System in detail down to the level of a single line of code or engineering equation. The Government project team approves all Type B5 Development Specifications produced by the Development Phase Contractor for each CPCI. In addition, the Government project team participates in the evaluation of and exercises selection approval of sub-contractors for CPCI development. The Government will prepare to assume full responsibility for the System during the Maintenance Phase. The Government will finalize requirements for, and sponsor acquisition of, experimental data necessary to determine CPCI and System level accuracy.

Interfaces with the Government/industry user community commence with the Validation Phase, which begins when the First Level Release is provided to the users. The objectives of the Validation Phase are to establish within the Government/industry user community an operational capability with the System, contribute to the validation of the accuracy and operating cost of the System, and provide inputs from the user community to the Development Phase Contractor and the Government project team to maximize user orientation of the System during the remainder of the Development Phase and during the Maintenance Phase.

Helicopter manufacturers under contract to the Government validate the applicability of the System to their helicopter types by correlation with experimental data. (These contracts are separate from the Development Phase contract.)

Also during the Validation Phase, these helicopter manufacturers, along with Government users,

- Achieve an operational capability with the System
- Apply the System to current rotary-wing research and development efforts, in parallel with other methods of analysis, to evaluate the effectiveness of the System
- Identify minor errors and deficiencies, determine corrective measures, and recommend their implementation
- Identify areas where the System can be enhanced and make recommendations to the Government project team that the System be modified

It is expected that information associated with these activities will be transmitted to the Development Phase Contractor by the Government. Figure 36 summarizes the formal and informal lines of communication among the organizations involved.

#### 5.1.2 Project Organization

One of the most important and critical characteristics of the Project organization is the organizational capability to manage and carry out tasks within the work breakdown structure (WBS) that it is assumed will be a part of the Statement of Work of the Development Phase Request for Quotation. Although the tasks, subtasks, and lower level work elements of the WBS are not presently known, sufficient information has been provided by the Government for CSC to propose a Development Phase Project organization effectively structured to meet the anticipated tasks and subtasks in the Statement of Work. This project organization is presented in Figure 37.

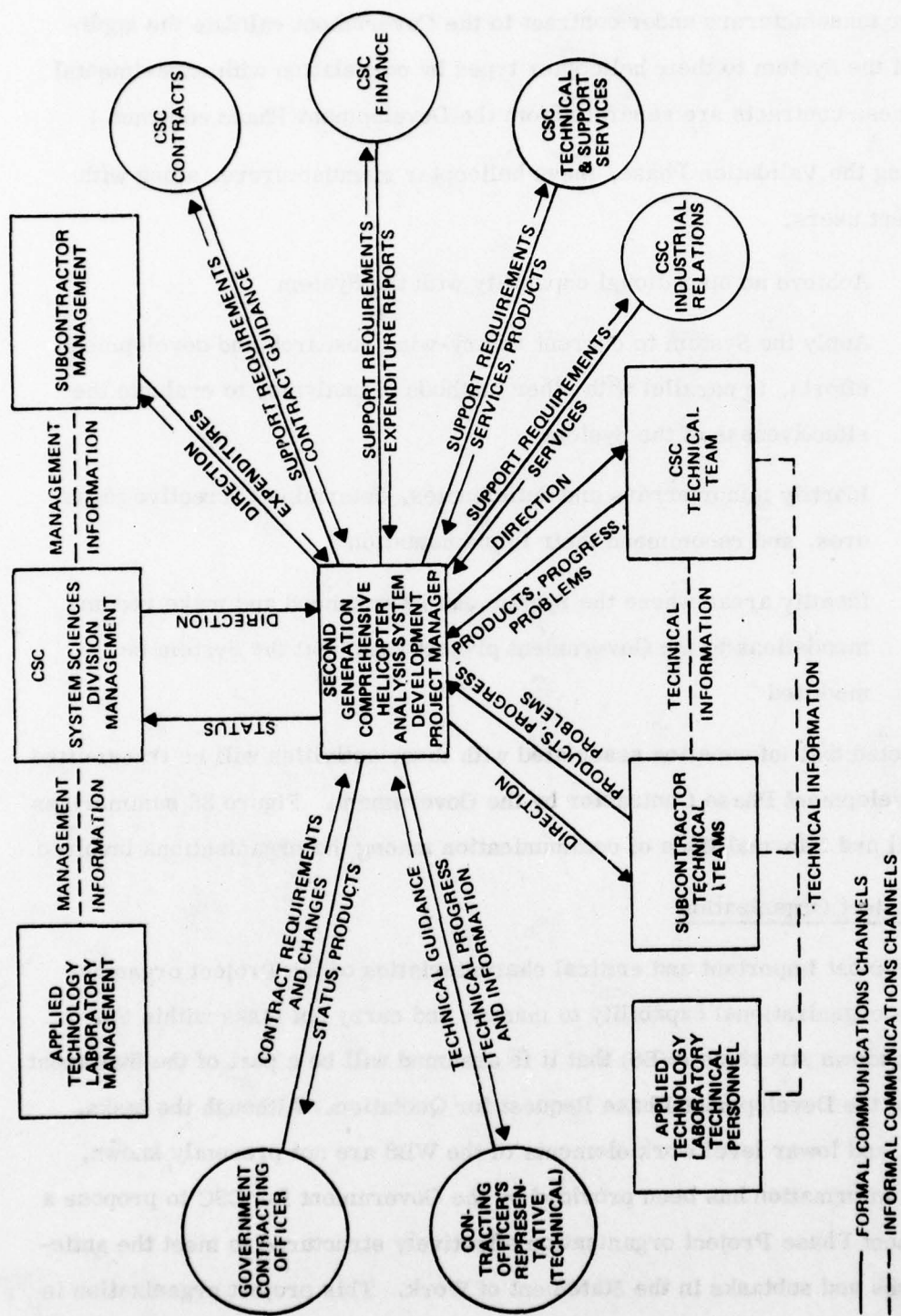


Figure 36. Lines of Communication During the Development Phase



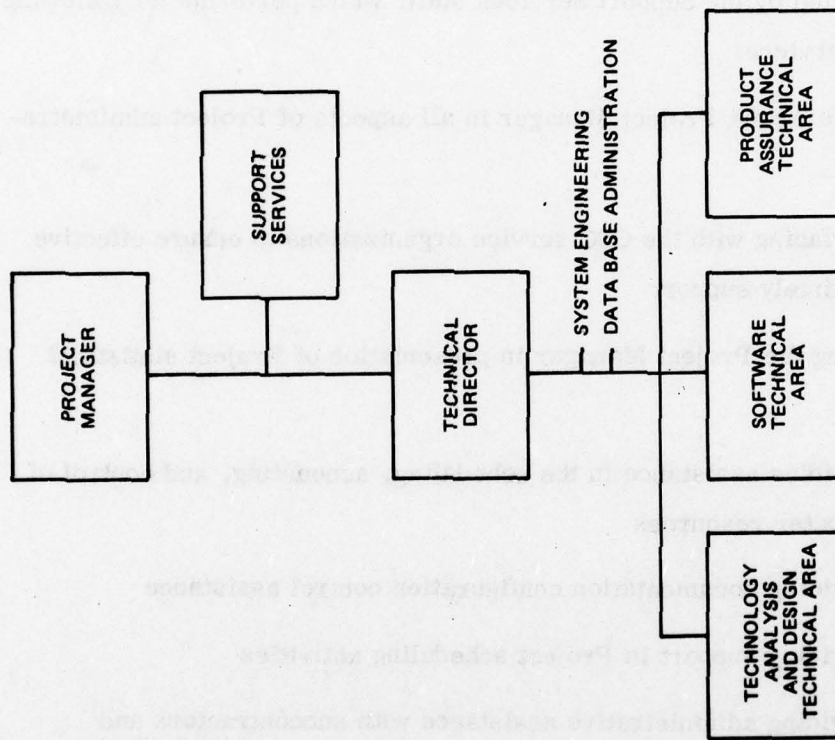


Figure 37. Preliminary Development Phase Project Organization

The Project Manager's objective is to complete the Project in accordance with the Statement of Work, the specifications, the Contract Data Requirements List, and other provisions of the contract and within the budgetary and time constraints of the contract. In other words, the Project Manager is completely responsible for the technical and financial success of the Project. A detailed discussion of the Project Manager's derived responsibilities and authorities that are corollaries to this primary responsibility is found in Section 5.1.2.1. The Project Manager is assisted by the Support Services staff, which performs the following administrative services:

- Assisting the Project Manager in all aspects of Project administration
- Interfacing with the CSC service organizations to ensure effective and timely support
- Aiding the Project Manager in presentation of Project statistical data
- Providing assistance in the scheduling, accounting, and control of computer resources
- Providing documentation configuration control assistance
- Providing support in Project scheduling activities
- Providing administrative assistance with subcontractors and vendors

The Technical Director has two primary responsibilities: to ensure the technical quality of the Project's technical products (source, object, and executable code and documentation) and to act with the full authority of the Project Manager should the Project Manager be absent. The responsibilities and authorities of the Technical Director are described in more detail in Section 5.1.2.2.

Reporting to the Technical Director are three line organizations: the Technology Analysis and Design Technical Area, the Software Technical Area, and the Product Assurance Technical Area. The primary responsibility of the Technology Analysis and Design Technical Area is to produce that part of the Type B5 Development Specifications associated with helicopter and mathematical analysis (the Technology Component CPCIs of the Operational Complex; see Section 2.3.1). Other responsibilities of the Technology Analysis and Design Technical Area are to provide documentation and training for the engineering user and methods developer, an analysis of System change requests during the Validation Phase, and an analysis of the effect of those proposed changes related to helicopter analysis and mathematical analysis. Most of the personnel in the Technology Analysis and Design Technical Area are drawn from the integrated team member subcontractor. No coding is performed by personnel of the Technology Analysis and Design Technical Area.

The Software Technical Area prepares that part of the Type B5 Development Specifications associated with executive and support software (these include the Executive Component of the Operational Complex and the Support Complex; see Sections 2.3.2 and 2.4, respectively). For those CPCIs that have been allocated to the Development Phase Contractor, this technical area is responsible for the following activities:

- Prepare complete Type C5 Computer Program Product Specifications
- Code all software in conformance with the approved design and Project programming standards
- Perform module and CPCI testing (Internal Developer Testing)

The Software Technical Area is also responsible for maintenance of the First Level Release and all associated software (i. e., non-engineering user) documentation. Because of this maintenance responsibility, the Software Technical Area will produce and maintain the System Maintenance Manual.



**The Product Assurance Technical Area is the quality assurance agent of the Technical Director. The responsibilities of this technical area are to**

- Conduct integration testing (Preliminary Qualification Testing)
- Conduct acceptance tests (Formal Qualification Testing)
- Conduct functional demonstration of the System
- Provide quality assurance and configuration management functions
- Integrate, and provide configuration management control for, the products of the Technology Analysis and Design and Software Technical Areas
- Maintain the source, object, and executable decks in a Program Support Library
- Prepare the products (e. g. , tapes) for the First and Second Level Releases
- Install and qualify the First and Second Level Releases
- Evaluate System performance
- Prepare the integration and acceptance test plans and specifications
- Prepare the integration and acceptance Test Analysis Reports
- Perform an independent quality assurance review of all technical documentation items (those CDRL items that are not associated with planning, progress, costs, and reviews) for technical content and conformance to standards
- Evaluate source code for conformance to Project programming standards
- Integrate into the System and qualify CPCIs developed by CPCI sub-contractors and the Government

- Provide the focal point for System maintenance
- Provide the focal point for communication with the Government/  
Industry User Community

In addition to these three organizations, a small technical staff reports to the Technical Director, providing the functions of system engineering and data base administration. The system engineering staff function provides a focal point for hardware requirements considerations and for the allocation of System functions to hardware or software. The data base administration function provides for the integrity of the Master Data Base.

The allocation of these line and staff functions to the Development Phase Contractor, the integrated team member subcontractor, and CPCI subcontractors is discussed in Section 5.1.2.3.

#### 5.1.2.1 Project Manager's Responsibilities and Authorities

The success of the development of the Second Generation Comprehensive Helicopter Analysis System will depend to a great extent on the effectiveness of the Project Manager. Based on extensive past experience with similar software development efforts (large, multiuser, transportable systems to be used by a large user community over a long period of time), CSC has found that the Project Manager must be completely responsible for the technical and financial success of the Project. His objective is to complete the Project in accordance with the contractual specifications and within the budgetary and time constraints of the contractual agreement. He exercises positive management controls as prescribed by CSC's policies and procedures and he is responsible for all determinations regarding cost, schedule, scope of work, and technical approach. Accordingly, he has the responsibility and authority for planning, organizing, conducting, directing and controlling, reviewing, and reporting all aspects of the Project (i.e., technical, schedule, administrative, and cost) from inception through successful completion, in accordance with the contract scope of work.

The Project Manager's general responsibilities are to define the contractual effort; assign responsibilities; and plan, schedule, budget, and authorize the work. He compares planned and actual costs, analyzes variances, incorporates changes, and develops estimates of final costs. Other responsibilities are to segregate, review, track, control, and report resource utilization to both CSC and Government management and anticipate situations that affect schedules and resources before-the-fact and provide alternative solutions. He is also responsible for establishing and monitoring all subcontracts.

To fulfill these responsibilities, the Project Manager

- Ensures the quality of all deliverable items
- Acts as the single point of contact between the Government and CSC
- Plans, evaluates, controls, and directs the schedule, cost, and technical performance of the Project
- Provides clear statements of work to the subcontractors
- Monitors subcontractor work to ensure performance, schedule, and cost compliance
- Develops, issues, maintains, updates, and controls all work schedules and plans
- Reviews and evaluates (on a regular basis) technical, cost, and schedule performance against plan
- Monitors performance on all tasks
- Controls all direct charges incurred on the Project
- Discusses problem areas with the Contracting Officer's Representative (Technical) when they occur and provides alternative solutions



- Identifies, describes, and establishes channels for Project communications
- Determines the need for and defines operating procedures peculiar to the Project's requirements
- Reports Project status to the Government in accordance with the specified contractual reporting requirements; reports internally to management of CSC's System Sciences Division
- Utilizes, through the Division's support organizations, CSC business management and data processing systems, facilities, and services to obtain maximum responsiveness to the Project's needs
- Determines the functional skill requirements and selects and manages all personnel assigned to the Project
- Maintains active liaison and open communication links with the Government, subcontractors, and the Government/Industry User Community to ensure complete visibility of Project status and to ensure CSC's responsiveness to the Government's needs
- Conducts regularly scheduled Project reviews for CSC management
- Meets regularly with Government personnel to review status, plans, and approaches

CSC delegates complete authority for Project management and control to the Project Manager. The Project Manager is authorized to

- Deal directly with the Government for all technical direction and input necessary for successful technical and financial performance of contractual requirements
- Assign, to CSC personnel and subcontractors, responsibilities for work and authorize work to be done within the contractual scope

- Incorporate changes and develop estimates of final costs
- Grant final CSC approval on all Project work
- Control assignments of CSC personnel to the Project
- Authorize and approve all labor and other direct expenditures charged to the contract

#### 5.1.2.2 Technical Director's Responsibilities and Authorities

The Project Manager delegates to the Technical Director all the necessary authority to fulfill the latter's primary responsibility, which is to ensure the high quality of all Project source code and technical documentation. In addition, the Project Manager delegates to the Technical Director all the authorities of the Project Manager in the Project Manager's absence. The Technical Director is the immediate supervisor of the three Technical Area Leaders and therefore has the necessary authority to ensure high-quality deliverables.

#### 5.1.2.3 Line Organizations

The three line organizations reporting to the Technical Director employ personnel from the Development Phase Contractor and the integrated team member subcontractor. In addition, for those CPCIs developed by the Government and CPCI subcontractors, it is convenient to consider Government personnel and CPCI subcontractor personnel as belonging to the Software Technical Area. The following is a summary of the work performed by persons of the different organizations within the three line organizations:

- Most of the personnel in the Technology Analysis and Design Technical Area are affiliated with the integrated team member subcontractor.
- Personnel in the Technology Analysis and Design Technical Area act as consultants, as required, to personnel in the Software Technical

Area in the module and CPCI testing of the software of the Technology Component.

- There are no Government personnel\* or CPCI subcontractor personnel in the Technology Analysis and Design Technical Area.
- The majority of the personnel in the Software Technical Area are affiliated with the Development Phase Contractor. At least one full-time person affiliated with the integrated team member subcontractor aids in internal developer testing at the module and CPCI levels. The Development Phase Contractor personnel in the Software Technical Area act as consultants to Government and CPCI subcontractor personnel in the same technical area in the interpretation of Project standards. (This does relieve these latter personnel from the responsibility of knowledge and implementation of the standards.)
- The majority of the personnel in the Product Assurance Technical Area are affiliated with the Development Phase Contractor. However, a significant contribution from integrated team member subcontractor personnel is required to conduct tests, prepare test documentation, and analyze System change requests. CSC does not envision any Government personnel or CPCI subcontractor personnel in the Product Assurance Technical Area.

Table 4 explains the relationships among the elements of the Project organization (Project Manager, Support Services Staff, Technical Director, Technical Director's Staff, Technology Analysis and Design Technical Area, Software Technical Area, and Product Assurance Technical Area), the activities they perform, and the organizations involved and the extent of their involvement.

---

\*The term "Government personnel" is used in this section to mean those organizations who are producing Government-furnished CPCIs.



**Table 4. Relationships Among the Organizations Involved in System Development and Maintenance (1 of 2)**

INTERNAL PROJECT ORGANIZATIONAL ELEMENT	PROJECT ACTIVITY	ACTIVITY TYPE	MAJOR ORGANIZATIONAL RESPONSIBILITIES						NOTES
			DEVELOPMENT PHASE CONTRACTOR	INTEGRATED MEMBER SUB CONTRACTOR	CPCI SUB CONTRACTORS	GOVERNMENT CPCI DEVELOPER	GOVERNMENT/INDUSTRY USER COMMUNITY	GOVERNMENT PROJECT OFFICE	
PM	PROJECT MANAGEMENT	NE	D					RA	
PM	PREPARE MANAGEMENT PLANS	E	D					RA	
PM, TD	PREPARE TECHNICAL PLANS	E	P	S				RA	
PM, TD	PREPARE CPCI DEVELOPMENT PLAN	E	P	S				RA	
SS	SUPPORT SERVICES	NE	P	S					
TD	TECHNICAL DIRECTION	NE	D						
TD	SYSTEM ENGINEERING	NE	P	S			A		2
TD	DATA BASE ADMINISTRATION	NE	P	S			A		2
PA	PREPARE DESIGN AND PROGRAMMING STANDARDS	E	P	S				RA	
TAD, SW	PREPARE TYPE BS DEVELOPMENT SPECIFICATIONS FOR THE TECHNOLOGY COMPONENT	E	S	P				RA	2
SW	PREPARE TYPE BS DEVELOPMENT SPECIFICATIONS FOR THE EXECUTIVE COMPONENT AND SUPPORT COMPLEX	E	P	A				RA	2
SW	PREPARE TYPE CS PRODUCT SPECIFICATIONS	E	P	S		D		RA	3
PA	REVIEW ALL TECHNICAL DOCUMENTATION FOR INTEGRITY, COMPLETENESS, AND CONFORMANCE TO STANDARDS (INCLUDES TYPE BS AND TYPE CS SPECIFICATIONS)	NE	P						
PA, TAD	PREPARE THE TEST AND IMPLEMENTATION PLANS	E	P	S		D		RA	
TAD, PA, SW	PREPARE THE USERS MANUAL	E	S	P		D		RA	2, 3
SW	CODE ALL SOFTWARE IN CONFORMANCE WITH PROJECT PROGRAMMING STANDARDS	E	D			D		RA	3, 4
SW, TAD	PERFORM MODULE AND CPCI TESTING (INTERNAL DEVELOPER TESTING)	E	P	S		D		R	3, 4
PA	PROVIDE CONFIGURATION MANAGEMENT FUNCTIONS	NE	D					R	

\*KEY A = ADVISORY RESPONSIBILITY  
D = DELEGATED RESPONSIBILITY  
E = END-ITEM ORIENTED  
IC = INFORMAL COMMUNICATION  
NE = NON-END-ITEM ORIENTED  
P = PRIMARY RESPONSIBILITY  
PA = PRODUCT ASSURANCE TECHNICAL AREA  
PM = PROJECT MANAGER  
R = REVIEW (INFORMAL DELIVERABLES)  
RA = REVIEW AND APPROVE (FORMAL DELIVERABLES)  
S = SUPPORT RESPONSIBILITY  
SS = SUPPORT SERVICES  
TAD = TECHNOLOGY DESIGN AND ANALYSIS TECHNICAL AREA  
TD = TECHNICAL DIRECTOR

- NOTES 1. THE TERM "GOVERNMENT CPCI DEVELOPER" IS USED HERE TO MEAN THOSE ORGANIZATIONS PRODUCING GOVERNMENT FURNISHED CPCI.  
2. ADVISORY SUPPORT FROM THE GOVERNMENT/INDUSTRY USER COMMUNITY IS INFORMAL AND IS TRANSMITTED TO THE DEVELOPMENT PHASE CONTRACTOR BY THE GOVERNMENT.  
3. THE DEVELOPMENT PHASE CONTRACTOR PREPARES ALL TYPE CS PRODUCT SPECIFICATIONS EXCEPT FOR THOSE CPCI EXPLICITLY DELEGATED CONTRACTUALLY TO CPCI SUBCONTRACTORS AND/OR DESIGNATED AS BEING FURNISHED BY THE GOVERNMENT.  
4. ALL SOFTWARE SHALL BE CODED IN CONFORMANCE TO THE APPROVED DESIGN AND PROJECT PROGRAMMING STANDARDS INDEPENDENT OF THE ORGANIZATION PRODUCING THE SOFTWARE.  
5. EVEN THOUGH CPCI SUBCONTRACTORS AND THE GOVERNMENT ARE RESPONSIBLE FOR ENSURING CONFORMANCE TO PROGRAMMING STANDARDS (SEE NOTE 4), THE DEVELOPMENT PHASE CONTRACTOR SHALL VERIFY THIS CONFORMANCE BEFORE INTEGRATING SUCH CPCI INTO THE SYSTEM.

Table 4. Relationships Among the Organizations Involved in System Development and Maintenance (2 of 2)

INTERNAL PROJECT ORGANIZATION ELEMENT*	PROJECT ACTIVITY	ACTIVITY TYPE*	MAJOR ORGANIZATIONAL RESPONSIBILITIES*					GOVERNMENT PROJECT OFFICE	NOTES
			DEVELOPMENT PHASE CONTRACTOR	INTEGRATED TEAM MEMBER SUB-CONTRACTOR	CPIC CONTRACTORS	GOVERNMENT DEVELOPER	GOVERNMENT/INDUSTRY/USER COMMUNITY		
PA	PROVIDE THE SOFTWARE INTEGRATION FUNCTION	NE	D						
PA	MANAGE THE PROGRAM SUPPORT LIBRARIES	NE	D					RA	3.5
SW, PA	EVALUATE SOURCE CODE FOR CONFORMANCE TO PROGRAMMING STANDARDS	E	D						
SW, PA	DETERMINE SYSTEM PERFORMANCE	E	D						
PA	CONDUCT INTEGRATION (PRELIMINARY QUALIFICATION) TESTS	E	P	S				R	3
PA	CONDUCT ACCEPTANCE (FORMAL QUALIFICATION) TESTS	E	P	S				RA	
TAD, SW	ANALYZE PROBLEMS OCCURRING DURING INTEGRATION AND ACCEPTANCE TESTING	NE	S	P					
PA	PREPARE THE TEST ANALYSIS REPORTS	E	P	S				RA	
SW, PA	PREPARE THE SYSTEM MAINTENANCE MANUAL	E	D				A	RA	
PA, SW, TAD	PREPARE AND CONDUCT TRAINING	E	P	S			A	RA	
PA	PREPARE RELEASE PRODUCT SETS	E	D				A	RA	
PA	INSTALL AND QUALIFY THE FIRST LEVEL RELEASE	E	P	S			A	S, RA	
PA	INSTALL AND QUALIFY THE SECOND LEVEL RELEASE	E	S	S			A	P, RA	
PA	INTEGRATE INTO THE SYSTEM AND QUALIFY RELEASES OF THE FIRST AND SECOND LEVEL SUBCONTRACTORS AND THE GOVERNMENT	E	P	S			A	R	
PA	PROVIDE THE FOCAL POINT FOR SYSTEM MAINTENANCE	NE	D					S, RA	
SW, TAD	ANALYZE IMPACT OF SYSTEM CHANGE REQUESTS	NE	P	S					
PA	IMPLEMENT APPROVED SYSTEM CHANGE REQUESTS	E	P	S					
PA	PROVIDE THE FOCAL POINT FOR COMMUNICATION WITH THE GOVERNMENT/INDUSTRY USER COMMUNITY	NE	D				IC		

\*KEY: A - ADVISORY RESPONSIBILITY  
D - DELEGATED RESPONSIBILITY  
E - END-ITEM ORIENTED  
IC - INFORMAL COMMUNICATION  
NE - NON-BRO-ITEM ORIENTED  
P - PRIMARY RESPONSIBILITY  
S - SUPPORT SERVICES

PA - PRODUCT ASSURANCE TECHNICAL AREA  
PM - PROJECT MANAGER  
R - REVIEW (FORMAL DELIVERABLES)  
RA - REVIEW AND APPROVE (FORMAL DELIVERABLES)  
S - SUPPORT RESPONSIBILITY  
SS - SUPPORT SERVICES

SW - SOFTWARE TECHNICAL AREA  
TAD - TECHNOLOGY DESIGN AND ANALYSIS TECHNICAL AREA  
TD - TECHNICAL DIRECTOR

NOTES: 1. THE TERM "GOVERNMENT CPIC DEVELOPER" IS USED HERE TO MEAN THOSE ORGANIZATIONS PRODUCING GOVERNMENT-FURNISHED CPIC.  
2. ADVISORY SUPPORT FROM THE GOVERNMENT/INDUSTRY USER COMMUNITY IS INFORMAL AND IS TRANSMITTED TO THE DEVELOPMENT PHASE CONTRACTOR BY THE GOVERNMENT.  
3. THE DEVELOPMENT PHASE CONTRACTOR PREPARES ALL TYPE OS PRODUCT SPECIFICATIONS EXCEPT FOR THOSE CPIC EXPLICITLY DELEGATED, CONTRACTUALLY, TO CPIC SUBCONTRACTORS AND/OR DESIGNATED AS BEING FURNISHED BY THE GOVERNMENT.  
4. ALL SOFTWARE SHALL BE CODED IN CONFORMANCE TO THE APPROVED DESIGN AND PROJECT PROGRAMMING STANDARDS INDEPENDENT OF THE ORGANIZATION PRODUCING THE SOFTWARE.  
5. EVEN THOUGH CPIC SUBCONTRACTORS AND THE GOVERNMENT ARE RESPONSIBLE FOR ENSURING CONFORMANCE TO PROGRAMMING STANDARDS (SEE NOTE 4), THE DEVELOPMENT PHASE CONTRACTOR SHALL VERIFY THIS CONFORMANCE BEFORE INTEGRATING SUCH CPIC INTO THE SYSTEM.

Each activity is classified as end-item oriented (E) or non-end-item oriented (NE). The responsibility for performing an activity can be one of three types: delegated (D), support (S), or advisory (A). An organization is delegated an activity without any support or advice from another organization when the nature of the activity warrants it. It is CSC's goal to delegate as many activities as possible to decrease the probability of communication gaps and to decrease travel and per diem costs. To help meet this goal, CSC strives to make an activity end-item oriented whenever possible. When an activity cannot be delegated completely to one organization, a principal (P) organization will be delegated primary responsibility, if appropriate for the activity, and other organizations will be delegated secondary/support (S) responsibility or tertiary/advisory (A) responsibility.

## 5.2 MANAGEMENT PLANS

The management plans in this section consist of the Development Control Plan (Section 5.2.1); the Work Management Plan (Section 5.2.2); the Communication Plan (Section 5.2.3); the Internal Review and Reporting Plan (Section 5.2.4); the Configuration Management Plan (Section 5.2.5); and the Documentation Management Plan (Section 5.2.6). These plans are based on similarly titled plans in the Baseline Development Plan; the plans presented here are for the most part summarized from those in the Baseline Development Plan.

### 5.2.1 Development Control Plan

Effective, positive control of the development of a large, complex software system is critical to its integrity and to its acceptance by its intended users. Control of the development of the Second Generation Comprehensive Helicopter Analysis System is particularly critical because the System is projected to be used for 15 years after the start of the Development Phase and because many of the CPCIs will be developed by different organizations.



To help ensure that the objectives for the System are met and the Government's premises for the Development Phase are followed, the following elements for controlling development are recommended:

1. The Government should require the Development Phase Contractor to prepare a System Development Plan for Government approval 2 months after award of the Development Phase contract. CSC envisions that the System Development Plan evolves from the Baseline Development Plan; defines the responsibilities of the various organizations involved in development; serves as an up-to-date baseline source of information about all technical and management aspects of the System's development; and is available to all concerned organizations. In addition to the up-to-date information presented in the Baseline Development Plan, the System Development Plan contains: a detailed technical approach that is the contractor's work breakdown structure response to the Development Phase contract's Statement of Work (see paragraph 5 below); a Source Selection Plan for governing the selections of CPCI subcontractors; a PERT chart for the entire Development Phase; a Project Milestone Schedule; a Project glossary; an Organizational Interface Plan; and a summary of the initial System design.

2. The Government should require the Development Phase Contractor to review, for consistency, accuracy, and completeness, the two sets of standards to be provided by the Government for the System: Programming Standards for the Second Generation Comprehensive Helicopter Analysis System<sup>35</sup> and Nomenclature of the Second Generation Comprehensive Helicopter Analysis System.<sup>36</sup> CSC suggests that the results of this review, in the form of recommended revisions to each standards document (i.e., the nomenclature document and the programming standards document), be identified as a CDRL data item to be submitted for Government approval 2 months after the award of the Development Phase contract. The approved recommended revisions should then be used as the basis for establishing the baselines of each standards document. The baseline standards documents can then serve as the up-to-date baseline source of nomenclature and

programming standards for all organizations involved in the development of the System.

3. The Government should require quarterly or triannual review meetings with the Government, the Government/Industry Working Group, the Technical Advisory Group, the Development Phase Contractor, the integrated team member subcontractor, CPCI subcontractors, and organizations providing Government-furnished CPCIs. These meetings will provide a forum for exchanging ideas and information and for solving problems between organizations. The format of these meetings would be described in the Organizational Interface Plan of the System Development Plan.

4. Methods should be recommended for providing regular and rapid communication among all organizations involved in development. Possible methods include a monthly newsletter supplemented by bulletins as required and a computer information network.

5. The System Development Plan should contain the detailed technical approach to be followed by the Development Phase Contractor, subcontractors, and the Government during the Development Phase. The order of presentation of the detailed technical approach is hierarchical and is keyed to the work breakdown structure (WBS), an integral part of the response to the Statement of Work in the Request for Quotation for the Development Phase. The detailed technical approach is presented at four successively more refined levels of detail, each level of detail corresponding to a level of the WBS:

- Project Level--Level I of the WBS
- Task Level--Level II of the WBS
- Subtask or Product Level--Level III of the WBS
- Activity or Subproduct Level--Level IV of the WBS

At the Project level, the objectives of the Project are presented, the tasks comprising the Project are named, and a brief statement of the objective of each task is provided.

At the task level, the scope and objectives of each task and the general approach to be followed in meeting the objectives of the task are presented. The scope is a brief summary of the task, the objectives are presented in terms of the end items (products) resulting from a successful conclusion to the task, and the general approach discussion emphasizes (1) the interrelationships of the task to other tasks and dependencies of the task on other tasks and (2) interrelationships and dependencies among the subtasks of the task.

At the subtask or product level, the inputs, activities, and product of each subtask are summarized. The interrelationships with, and dependencies on, other subtasks are discussed. The schedule and a summary of the resources budgeted for the subtask are presented. The standard that will be used to measure the quality of the product is referenced. The plan for integration of the subproducts is presented. This plan will emphasize (1) the definition of the interfaces among the organizations (e.g., subcontractors) responsible for the subproducts of the product and (2) the interfaces among the subproducts.

At the activity or subproduct level, the inputs, work elements, and results of each activity are presented. The schedule and resources budgeted for the activity are also given.

In addition to the narrative described above, the detailed technical approach is summarized in a Project Milestone Schedule and a PERT chart.

#### **5.2.2 Work Management Plan**

The Work Management Plan, presented in the Baseline Development Plan and summarized here, gives CSC's approach to managing the work in the Development Phase. The plan presents CSC's approach to (1) defining the work and assigning responsibilities to perform the work; (2) planning, scheduling, budgeting, and



authorizing the work; (3) controlling costs; (4) incorporating changes and developing final costs; (5) segregating, reviewing, tracking, controlling, and reporting resource utilization; (6) anticipating situations that affect schedules and resources; and (7) managing subcontractors. The paragraphs below summarize the main points of the Work Management Plan.

#### 5.2.2.1 Defining the Contractual Effort and Assigning Responsibilities

CSC's approach to defining the contractual effort of the Development Phase and assigning responsibilities has been and will be a continuing effort. This approach has been applied during the Predesign Phase and will be applied during the pre-proposal phase, and the proposal phase and will be finalized in the first 2 months of the Development Phase to arrive at a Detailed Technical Approach. The Detailed Technical Approach will include a detailed definition of the work in the form of a Work Breakdown Structure (WBS), and an accompanying narrative assignment of work, planned schedules, and budgets. This has already been discussed in paragraph 5 of Section 5.2.1. Immediately following award of the Development Phase contract, CSC proposes to meet with the Government to review this approach and identify aspects needing modification based on the Government's evaluation. A minor planning adjustment activity is anticipated to result from this meeting; other than this adjustment, CSC's management approach for defining the Development Phase effort and assigning responsibilities will have been implemented. The Detailed Technical Approach is documented in the System Development Plan.

Responsibilities will be assigned by CSC to itself, the integrated team member subcontractor, CPCI subcontractors, and the Government, based on the type of work to be done. All such assignment of responsibilities will of course be subject to Government approval. A discussion of what types of work CSC plans to assign to itself, the integrated team member subcontractor, its CPCI subcontractors, and the Government is provided in Section 5.1 of this report.

#### 5.2.2.2 Planning, Scheduling, Budgeting and Authorizing the Work

A basic part of the management process is the thorough planning of all elements of work. This involves the evolution of detailed development schedules for all tasks and subtasks; detailed budgets for all human resources and for subcontract and other direct cost items; and schedules for the delivery of all specifications and documentation to be produced. This section describes the development of the plans, schedules, and budgets that are the products of the work-planning activity. Work planning for the Development Phase began with pre-proposal work for the Predesign Phase; has continued during the Predesign Phase study efforts; will become more detailed during the proposal for the Development Phase; and will culminate with the System Development Plan approved by the Government 2 months after Development Phase contract award.

The first step in the effective control of schedules and costs is the planning activity, the results of which are incorporated in the System Development Plan and which provide the framework and measurement basis for monitoring progress. The initial part of the planning process is work definition, discussed in Section 5.2.2.1. The next activity involves specifying initiation and completion dates that are consistent with constraints imposed by the interdependencies among the subtasks and various other factors. These include the availability of required input data and requirements from the Government and industry interfaces, delivery dates specified in the contract, and efficient use of resources.

The first step in this process is to estimate the amount of time required to complete each subtask. Generally, this time is a function of the magnitude of the work required, the human resources assigned, the skill level and related experience of the assigned personnel, and the usefulness of computer time. The initial estimate is based on those human resource and skill levels judged to be most cost effective. The subtasks are then arranged in a schedule that accounts for interdependencies, specified dates for reviews, and use of parallel activities to maintain continuity of personnel and uniformity of workloads.

The initial schedule is then analyzed and revised as necessary to ensure that it meets the requirements. The resulting planned schedule is called the Project Milestone Schedule. After Development Phase initiation, the Project Milestone Schedule will be reexamined to take into account any changes resulting from initial discussion with the Government. This schedule will be a principal management control tool throughout the Development Phase. As the Development Phase evolves, the schedule will be reviewed frequently (at least weekly). It will be revised as necessary to include deviations from the planned schedule and more detailed schedule information when appropriate. Thus, planning and scheduling will be continuing activities throughout the Development Phase. Plans must be continually reviewed and revised to be responsive to the current situation and latest information available. These plans and the necessary revisions to them will be continually available to the Government to review, analyze, and provide feedback and direction.

When the schedule development process is complete, the associated human resource requirements are assembled for all subtasks. Computer time, travel, and other costs are determined, and estimates of support-services requirements as well as the time for project management and review activities are determined. This information is then developed into a planned budget which is the principal control tool for monitoring financial performance.

Work authorization is the responsibility of the Project Manager. After completion of the work definition activity, which defines the tasks and subtasks to the lowest practical level, and assignment of responsibilities for work as discussed in the previous section, the Project Manager authorizes commencement of subtasks in accordance with the Project Milestone Schedule. Deviations from the schedule are brought to the attention of the Project Manager by the internal reporting system described in Section 5.2.4.



Subcontractor work authorization is also the responsibility of the Project Manager, who reviews and approves each subcontractor statement of work. The subcontractor will be authorized to proceed after the subcontract has been accepted and signed by both parties and approved by the Government.

#### 5.2.2.3 Controlling Costs

For effective monitoring, analysis, and control of costs and other resource expenditures, the Project Manager must have accurate and up-to-date information. CSC's accounting system is designed to provide the needed data. This section describes how costs are controlled by use of cost and manpower information provided to the Project Manager.

The information provided to the Project Manager must convey both the amount expended on the total project in each of the cost accrual categories and the distribution of the expenditures among the tasks composing the project. CSC's time and cost accounting procedures provide the Project Manager with full control over resource expenditure and great flexibility in accumulating cost data in a variety of categories specifically tailored to the Project. Data are entered into CSC's automated accounting system, where they become part of a common data base from which expenditure reports for the Project Manager, CSC top management, and the Government are generated. All parties are thus ensured of consistent resource expenditure data.

#### 5.2.2.4 Incorporating Changes and Developing Final Costs

Effective control and monitoring procedures allow accurate assessment of changes in requirements or specifications and determination of the effects of such changes on final costs and schedules. In a program such as the Second Generation Comprehensive Helicopter Analysis System, controlling changes that affect the final outcome is an extremely important function of Project management. CSC will evaluate changes during the Development Phase based on the assumption that Section L, Paragraph 62, of CSC's Predesign Phase contract will also appear in the

Development Phase contract, and that the Configuration Management Plan will provide the policy for proposing and implementing changes. The reader is referred to the Baseline Development Plan for more details.

#### 5.2.2.5 Segregating, Reviewing, Tracking, Controlling, and Reporting Resource Utilization

CSC's accounting system allows detailed segregation, tracking, and reporting of costs and hours within predetermined categories and provides weekly and monthly reports of resource utilization for the Project Manager's review.

The data provided on the weekly Project Manager's Report and the weekly Labor Distribution Report is as follows:

- Expenditure of hours by named individuals
- Total hours expended for the week and total funds expended for the week by category (direct labor, overhead, etc.)
- Cumulative hours and funds expended to date
- Budgeted hours and funds
- Percentage of total hours and funds expended
- Variance from budgets

The percentage of the work actually completed, by work element, is estimated by the lead technical personnel of the particular tasks or subtasks and is reported on the Project Milestone Schedule maintained by the Project Manager.

General CSC management also has responsibility for tracking and controlling the Development Phase. To support fulfillment of this responsibility, each Project Manager is required to complete a Monthly Project Status Report (MPSR) to Division and Corporate management. This report contains a brief summary of task status with respect to performance level, conformance to

schedule, and resource expenditures. Problems in any of these areas are highlighted and discussed to alert management to the possible need for special attention. The MPSR also includes an up-to-date summary financial chart depicting budgeted expenditures versus time and actual expenditures to date and a current Project Milestone Schedule for the Project indicating actual-versus-planned progress.

#### 5.2.2.6 Anticipating Situations That Affect Schedules and Resources

No one wants to be surprised by "sudden" slips in schedule or increases in costs. Completion dates of major milestones do not slip several weeks during a single day; nor do costs, in resources or dollars, escalate overnight. If they appear to do so, it is almost totally due to a failure in management control and to a lack of visibility or communication. The majority of serious problems, delays, and cost increases can be completely avoided if effective control techniques are used to identify problems early enough to be remedied with corrective action.

The basic structure of the control process for schedules, costs, and other resource items which CSC currently uses and plans to use for the Development Phase consists of the following:

- **Planning--**Thorough planning of all elements of work, including detailed development schedules for all significant work elements, detailed and realistic budgets for all human resource and cost items, and schedules for all specifications and documentation. The key to successful planning, monitoring, and control is the breakdown of the work into discrete work elements whose completion can be objectively evaluated.
- **Monitoring and reporting--**Continual tracking of performance, progress, and costs and comparison of these items with the original plans and budgets; communication of the results of this monitoring both internally and to the Government through written and



verbal reporting and informal communication; and projection of possible problem areas and development of contingency plans.

- Corrective action--Quick response to identified problem areas, schedule variances, or deviations from budget through the use of management action, alternative solutions, or revised plans.

To be effective, the planning function must result in a detailed Project Milestone Schedule based on activities characterized by meaningful measures of progress and completion. For the Development Phase, these measures are linked to the documentation to be delivered, the reviews to be conducted, the source code to be produced, the tests to be conducted, the CPCIs to be developed, and the software to be released to the helicopter manufacturers and to the Government agencies. The monitoring and reporting functions are designed to collect and present information to management, permitting meaningful assessment of progress against plan. Finally, the information and planning must facilitate management's ability to evaluate the effect of alternative approaches aimed to correct deviations between plan and progress. The responsibility for collecting and evaluating this information rests with the Project Manager, as does the authority for specifying that it be provided. The Project Manager will be assisted in this function by all members of the technical staff.

The progress-monitoring activities to be used include weekly progress reports; regular staff meetings; internal design reviews; documentation reviews; test material reviews; source code reviews; and informal, person-to-person reviews. These activities are described in Section 5.2.4.

If a serious existing or potential problem is identified either by Project personnel or through any of the previously mentioned progress-monitoring activities, the Project Manager may call for a technical audit. To conduct this audit, an ad hoc team of highly qualified senior technical staff members will be formed

by the Project Manager. The participation of specialized experts from throughout the Division and, if necessary, other divisions of the Corporation, can be requested. This team will review the problem in detail, working closely with Project personnel; identify alternative solution approaches; and prepare a recommended course of action. The result of this audit will be provided immediately to the Project Manager. Responsibility for implementing the solution rests with the Project Manager.

Another key element of successfully anticipating situations that affect schedules and resources is careful, formalized control of changes to the System requirements. Experience demonstrates that uncontrolled modification of requirements, if permitted, leads rapidly to an uncontrolled state of escalating costs, degraded system integrity and reliability, and unknown status. The procedures embodied in the Configuration Management Plan in the System Development Plan will be applied to ensure that changes in the Development Phase are effectively controlled during System development.

#### 5.2.2.7 Managing Subcontractors

Success in any contract relationship is achieved through an explicit statement of work to be performed by the prime contractor; thorough understanding of the work to be performed by the subcontractor; periodic reports of progress measured against meaningful milestones; and frequent, effective formal and informal communications to continually refine and feed back mutual understanding.

The relationships between CSC and the integrated team member subcontractor and the CPCI subcontractors, will be governed by the work-definition and work-planning principles given in Section 5.2.2.7.1 and the work-monitoring principles in Section 5.2.2.7.2. These principles will be expanded into plans prior to the start of the Development Phase.

#### 5.2.2.7.1 Work-Definition and Work-Planning Principles

Defining and planning the work to be done by subcontractors are guided by the principles given below.

- The Government's Statement of Work in the prime contract is mapped as directly as possible into the Statement of Work for the subcontractor.
- As many of the work elements as possible are made end-item (deliverable) oriented to permit effective monitoring and to minimize travel and per diem costs.
- Subcontractor personnel work on site with CSC personnel during crucial stages of work planning and software integration when face-to-face communication is necessary.
- Deliverables are specified in detail to permit effective monitoring of progress versus plan.
- Deliveries of subcontract data items are provided incrementally.
- The provisions of the prime contract are passed down to each subcontract to the maximum extent applicable.
- A detailed performance plan, which will contain both technical and management subplans, is required of each subcontractor.
- Subcontractor personnel participate fully with CSC personnel during formal reviews with the Government and during meetings with the Government/Industry Working Group and the Technical Advisory Group.
- Procurements are competitive to the greatest extent possible.



#### 5.2.2.7.2 Work-Monitoring Principles

Formal written Monthly Letter Progress Reports and Monthly Cost and Performance Reports are required from all subcontractors. These reports plus review and approval or disapproval of contract deliverables establish one level of monitoring of the subcontractor by CSC. Moreover, because communication is essential to arrive at a satisfactory end-product, each subcontract is monitored by the Project Manager, who will communicate informally with the subcontractor's Project Manager on a regularly scheduled, weekly basis to determine the actual progress and to ensure that all work is proceeding according to plan. Problems are addressed immediately, and alternative solutions discussed. In addition to the regular weekly status telephone calls, frequent, usually daily, conversation takes place between CSC and subcontractor personnel, including conference-type telephone calls when required. In addition, to ensure effective communication and control, the CSC Project Manager makes personal visits to subcontractor facilities at least once every 2 months when all work is being performed exclusively at the subcontractor's facility.

#### 5.2.3 Communication Plan

The Communication Plan describes how the Development Phase Contractor will communicate with the Government and its subcontractors. A professional, disciplined approach for communicating and reporting all technical activities is prerequisite for project success. Accordingly, CSC plans to use formal technical communication for the transmission of planned activities, status and progress reports, and problems. This plan is based on the assumption that the CDRL will (and should) contain the same media for communication between CSC and the Government during the Development Phase as during the Predesign Phase. The Contract Performance Plan, monthly letter progress reports, monthly cost and performance reports, progress/status meeting reports, agendas and minutes of

design reviews, and interim technical reports are all vehicles for CSC's technical and financial communication with the Government.

Section 5.2.3.1 describes CSC's proposed approach for formally communicating and reporting all technical and financial information between the Government and CSC, and between CSC and its subcontractors. Section 5.2.3.2 describes an approach for establishing a convenient but controllable mechanism for informal technical communication between CSC and its subcontractors.

#### 5.2.3.1 Formal Communication

The Contract Performance Plan presents the Development Phase Contractor's approach to fulfilling all contract requirements. This Contract Performance Plan is a revised version of the plan submitted with the proposal for the Development Phase. Revisions are based on negotiations and Government feedback. As work proceeds on the development of the System, the Development Phase Contractor maintains regular communication with the Government to ensure that potential problems are dealt with as they arise. The reporting structure should, as a minimum, consist of the following:

- Letter Progress Reports. These monthly reports are the principal formal vehicle for reporting status, progress, problems, and planned activity to the Government. The Letter Progress Report contains brief statements on the status of the Project and provides a comparison of actual progress with the planned progress of the Contract Performance Plan. It discusses the schedule status of each subtask that was scheduled to begin, end, or continue in the reporting period. An assessment of the effect of existing problems on task performance is given. Specific plans for work scheduled for the next 2-month period and reports of meetings with the Government and industry, and their results, are provided.

- Monthly Cost and Performance Reports. These reports provide detailed information on expenditures of hours and funds. The hours expended

for each work element (e.g., task, subtask) are compared with the budgeted hours, and a judgement is made as to the adequacy of the remaining hours for completing the effort. Information is also provided about the funds expended for each work element, a comparison with the budgeted amount is reported, and a judgement is made as to the adequacy of funds for completion. In addition, the percentage of work completed is presented for each work element with an estimate of the cumulative percentage of total work completed to date.

- Progress/Status Meetings. CSC recommends regular meetings every month with the Government at Fort Eustis. This type of technical interchange has proven to be an invaluable method for reviewing progress and discussing areas of concern. Such meetings will afford the Government an opportunity to raise questions and receive whatever detailed information is required for all aspects of the ongoing activities. Following each meeting, the contractor will, in accordance with the CDRL, prepare minutes of the meeting to document the discussion. Recommendations made by CSC or the Government will be listed, and action items will be noted along with the due dates.

- Reports From Subcontractors. The prime contractor will be the sole reporter to the Government. Letter progress reports and cost and performance reports from subcontractors are incorporated with internal information and included in the monthly reports to the Government. (This is a corollary to the principle stated in Section 5.2.2.7.1 that provisions in the prime contract will be "passed down" to each subcontract.) Subcontractors provide a monthly progress report summarizing technical activities for the month and identifying any possible problem areas. Subcontractors also provide Monthly Cost and Performance Reports. Subcontracts will stipulate that these reports be consistent with the format of the prime contractor's report to the Government.

- Technical Reports. Technical reports include all the deliverable technical documents defined in the CDRL that are not categorized as status reports,



progress reports, or meeting minutes. Examples of these technical documents include Type B5 Development Specifications, Type C5 Product Specifications, the Data Base Specification, the Test and Implementation Plan, the Test Analysis Report, the User's Manual, and the System Maintenance Manual. The Development Phase Contractor is responsible for the format, content, and quality of each document whether the document is prepared by the integrated team member subcontractor, or by the CPCI subcontractor.

#### 5.2.3.2 Informal Communication

All managers recognize the importance of informal communication in meeting objectives and use informal communication to good advantage. Internal and external organization lines, responsibilities, lines of authority, and lines of communication between the Development Phase Contractor, its subcontractors, the Government, the Government/Industry Working Group, and the Technical Advisory Group have been discussed in Section 5.1. Informal communication exists for the purpose of coordination, advice, and support. Informal communication with Government/Industry Working Group members and Technical Advisory Group members occurs at the joint meetings proposed as part of the Development Control Plan discussed in Section 5.2.1. Informal communication among CSC and subcontractor personnel consists of the following:

- Trips to CSC's facility by subcontractor personnel and to subcontractors' facilities by CSC personnel
- Meetings to review work status
- Face-to-face contact during the workday
- Telephone conversations
- Informal memoranda regarding recommendations, technical data, guidance, information, etc.

- Informal exchange of preliminary versions or parts of documents (usually to speed information flow and later formalized for record purposes)

A straightforward, proven mechanism for controlling informal written communication--a set of numbered Project memoranda--is planned. These provide the means to quickly document and transmit technical information that will be useful to all Project personnel, independent of organizational affiliation, and also promote communication of technical matters among Project personnel. The memoranda are easily referenced, and a file is maintained. For future reference or formalization, other memoranda are kept of significant telephone conversations, document exchanges, visits, and meetings.

#### 5.2.4 Internal Review and Reporting Plan

Because of the complexity of the System and the large number of organizations involved, effective review and reporting procedures are important to overall Project success. Based on an analysis of the development environment and its past experience on projects of similar size and complexity, CSC recommends, in addition to the formal contractual communication vehicles for reporting and review proposed in Section 5.2.3, the following ways to enhance control within the Project: weekly activity reports, regular staff meetings, internal design reviews, documentation reviews, test material reviews, source code reviews, and informal reviews. These are discussed below.

- Weekly Progress Reports. Each week, Project personnel prepare a progress report discussing activities during the week, planned future activities, and problems encountered or anticipated. All problems must be accompanied by suggested alternative solutions. The Project Manager analyzes these weekly reports to measure reported technical progress against the planned progress documented in the Project Milestone Schedule and acts to resolve any problems identified.

- Regular Staff Meetings. At least once a week, the Project Manager conducts project review meetings with the Project Staff. The objectives of these meetings are to review status and problems as detailed in the Weekly Progress Reports, to identify and evaluate pertinent external developments and their potential impact, to discuss and resolve general project management problems (e.g., project communications, reporting, procedures), and to call attention to activities requiring special skills (e.g., short-term or quick-response application of senior technical specialists). These regular staff meetings are augmented with phone calls to subcontractors; conference-type calls are used when necessary.

- Internal Design Reviews. During design activities, management or designated technical personnel conduct internal design reviews by having designers "walk through" their designs. (The reviewers will evaluate the design and recommend improvements or modifications.) Project management directs action based on these recommendations. The use of design walk-throughs has proved to be highly effective in monitoring design status and quality and in satisfying needs for communication of design information among Project staff members.

- Documentation Reviews. All documentation products are submitted for detailed technical and quality control review. Technical review is the responsibility of the Technical Director or his delegated representative and considers such factors as conformance to specifications, accuracy, clarity, and completeness. (Quality assurance review is concerned with format, organization, clarity, language, and grammar.)

- Test Material Reviews. Following design activities, management or designated technical personnel conduct internal reviews of test material by having the developers of the material "walk through" the test specifications and procedures. (The reviewers will evaluate the planned tests and recommend additions or modifications). Only after test material has been reviewed and approved by designated personnel can the planned tests be executed. Experience



has demonstrated the effectiveness of testing walk-throughs in monitoring test status and in ensuring that the tests planned are sufficiently comprehensive to demonstrate the validity of both the design and the code.

- Source Code Reviews. Source code is one of the Development Phase Contractor's most important products. Therefore, software development monitoring procedures include walk-throughs of source code products by management and lead technical personnel to ensure accuracy and conformance with Project programming standards. This review of source code is a significant element of CSC's software development monitoring procedures. Each module must be formally reviewed and approved by designated personnel before it can be introduced into a System source library.

- Informal Reviews. In addition to the above formal monitoring procedures, CSC places a great deal of emphasis on informal methods. These methods rely on the ability of responsible senior technical and management staff members to work closely with less experienced staff members to check progress and provide daily guidance and encouragement in person. This informal method is highly productive because it monitors progress on a very detailed level and eliminates uncertainties that might exist about what progress is expected. A derivative benefit of this personal contact is the establishment of interpersonal rapport throughout the Project, creating an environment that contributes to the development and retention of a highly efficient, closely knit team.

#### 5.2.5 Configuration Management Plan

Configuration management is a discipline applying technical and administrative direction and surveillance to (1) identify and document the functional and physical characteristics of a configuration item, (2) control changes to those characteristics, and (3) record and report change-processing and implementation status. Configuration management is thus the means through which the integrity

and continuity of the design and various engineering and cost trade-off decisions are recorded, communicated, and controlled.

One of the most important aspects of configuration management is the concept of baseline management. A baseline is a document that formally identifies and establishes the initial configuration identification of a configuration item at specific times during its life cycle. A baseline provides a formal departure point for control of future changes in performance and design. Usually three baselines are defined: functional, allocated, and product. A functional baseline is usually a Type A System Specification, an allocated baseline is usually a Type B Development Specification, and a product baseline is usually a Type C Product Specification.

It is assumed that the Contract Data Requirements List (CDRL) of the Development Phase contract will require the Development Phase Contractor to submit a Configuration Management Plan to the Government.

CSC envisions the Configuration Management Plan for the development of the Second Generation Comprehensive Helicopter Analysis System to be in eight sections: Section 1, Organization; Section 2, Configuration Identification; Section 3, Configuration Control; Section 4, Configuration Status Accounting; Section 5, Subcontractor/Vendor Control; Section 6, Program Phasing; Section 7, Management Integration of Configuration Management; and Section 8, Configuration Audits.

Section 1, Organization, will summarize the Project organization; the configuration management organization and its relationship to the Project organization; the responsibilities and authorities of the configuration management function; the configuration management system; policies and procedures related to configuration management; program support library services; and technical library

services. The responsibilities of the configuration management function are as follows:

- Preparation of the Configuration Management Plan and, once approved by the Government, performance of audits to ensure conformance by the Project organization to the principles of the plan
- Review of all CDRL items to ensure conformance to contractual configuration management requirements
- Maintenance of all Project data
- Review of all configuration indexes for completeness and accuracy
- Preparation of Engineering Change Proposals (ECPs), Advanced Change/Study Notices (AC/SNs), and Specification Change Notices (SCNs)
- Review of all proposed changes, regardless of classification or origin, after a baseline has been established
- Development and implementation of working procedures that define the methods to be used for the identification and control of computer programs and support documentation

Section 2, Configuration Identification, will describe the process of defining and documenting the approved configuration of the System throughout its life cycle in terms of Type B5 Development Specifications and Type C5 Product Specifications. A specification tree will be used to show the relationship between and among the specifications. Specifications shall comply with the applicable military standards, e.g., MIL-STD-490, MIL-STD-483, and MIL-STD-480. Because of the multiorganization environment of the development of the System, interface control will be described in detail. The standards for assigning configuration identification numbers to specifications, CPCIs, CPCI copies, CPCI



versions, and magnetic tapes and other storage media will also be presented, as well as the procedures for release of the System.

Configuration control is the systematic evaluation, coordination, approval or disapproval, and implementation of proposed changes to an established baseline. Section 3, Configuration Control, will discuss the preparation of ECPs, AC/SNs, and SCNs in accordance with MIL-STD-480; the content of ECPs; maintenance of the Type B5 Development Specifications and Type C5 Product Specifications; and configuration control of computer program code (cards, tapes, etc).

Section 4, Configuration Status Accounting, will state the Development Phase Contractor's plans for application of configuration index and status accounting records for the Development Phase.

Section 5, Subcontractor/Vendor Control, will describe the methods CSC will use to ensure that subcontractor/vendor practices adhere to the Project's Configuration Management Plan. This section will include the methods CSC will use to determine subcontractor/vendor capability in the configuration management area.

Section 6, Program Phasing, will establish the major milestones for implementation of the Configuration Management Plan. These include the following:

- Establishing the Configuration Control Board
- Phasing for specification-program implementation, including specification maintenance
- Establishing each of the configuration identifications
- Establishing interface control agreements with subcontractors and the Government
- Establishing configuration index and status accounting procedures

Section 7, Management Integration of Configuration Management, will describe the integration of configuration management activities with other Project management activities. The relationship between configuration management at the CPCI level and its relationship to the work breakdown structure for control of work authorization and cost control will be defined and discussed. This section will also provide detailed information about the relationship between events critical to configuration management and schedule of the Project (sequencing of design reviews, releases, audits, etc.).

Section 8, Configuration Audits, will contain CSC's plans for conducting both functional and physical configuration audits.

#### 5.2.6 Documentation Management Plan

Providing thorough, high-quality documentation is a principal Government objective for the Second Generation Comprehensive Helicopter Analysis System. Such documentation does not simply happen--it must be well conceived, well planned, and well executed. Detailed planning for the documentation for the Development Phase has begun during the Predesign Phase with the Documentation Plan in Section 5.3.3. The importance CSC attaches to documentation is illustrated by the fact that the Detailed Technical Approach, which is a part of the System Development Plan, organizes the work around the principal document deliverables. CSC's management methodology for producing thorough, high-quality documentation emphasizes the following:

- Detailed planning and scheduling
- Standards for document production
- Management reviews of all deliverables
- Use of CSC's Technical Publications Department

These four topics are discussed in detail in the Baseline Development Plan .

### 5.3 TECHNICAL PLANS

The technical plans in this section are divided into major technical plans and subordinate technical plans. The major technical plans presented here are taken, by and large, intact from the Baseline Development Plan. The three major technical plans are the Quality Assurance Plan (Section 5.3.1), the Testing Plan (Section 5.3.2), and the Documentation Plan (Section 5.3.3). There are five subordinate technical plans: the Data Processing Facility Plan (Section 5.3.4.1), the Data Management Plan (Section 5.3.4.2), the System Installation and Release Plan (Section 5.3.4.3), the Training Plan (Section 5.3.4.4), and the Maintenance Plan (Section 5.3.4.5).

#### 5.3.1 Quality Assurance Plan

This section defines the quality assurance methodology to be used during the Development Phase for ensuring satisfactory design, implementation, and testing. Section 5.3.1.1 defines the methodology to be used. Section 5.3.1.2 presents the general management approach to be used to implement the methodology. Sections 5.3.1.3, 5.3.1.4, and 5.3.1.5 describe the application of the methodology to design, implementation, and testing efforts, respectively.

##### 5.3.1.1 Quality Assurance Methodology

The methodology for quality assurance of the System includes the use of the following:

1. Top-down development
2. HIPO (Hierarchy plus Input-Process-Output) for design presentation
3. Data flow diagrams
4. Structured "walk-throughs"
5. Definition and enforcement of programming standards
6. Development of test specifications before coding
7. Structured programming
8. Independent testing



Top-down development involves designing, coding, and testing the highest level modules of a system first, in contrast to the more common methodology used in the past--bottom-up development. The top-down approach has the benefits of giving the critical modules near the top of the module-calling hierarchy of the software system the most testing, thus providing earlier warning of problems with the interfaces between and among modules and groups of modules.

HIPO is an approach to functional specification and documentation of software designed in a top-down fashion. Each function is designed using an Input, Process, Output (IPO) diagram, which lists the input and output and specifies the processing to be carried out. A hierarchical visual table of contents diagram relates the IPO diagrams and shows not only the functions and subfunctions to be carried out but also the relationships between them. The hierarchical structure of HIPO is well suited to the presentation of a top-down, hierarchical design created by starting at the top and subdividing into increasingly lower levels of detail. HIPO has been used by CSC throughout the Predesign Phase.

Data flow diagrams are used to show how System-level data flows through the System from function to function or subfunction to subfunction. Data flow diagrams show the relationships between the software elements of the System design and the data to be processed by the System. A form of data flow diagrams is presented in Section 2.5 to illustrate System Command Sequences and data flow.

A structured "walk-through" is a review session in which the originator of the design material, the test material, or code "walks" the reviewers through the design, the test materials, or the code in a step-by-step fashion to simulate the function under investigation. The intent of a structured "walk-through" is to detect problems in development as early as possible, when they are least expensive to correct. The effectiveness of structured "walk-through" has been demonstrated repeatedly.

Programming standards not only affect the legibility and reliability of software-as-written but also affect design concepts. The effectiveness of programming standards is proportional to their quality and management's commitment to their enforcement. Therefore, programming standards (in the form of the Programming Standards for the Second Generation Comprehensive Helicopter Analysis System<sup>35</sup>) will be finalized early and enforced rigorously throughout the Development Phase. The Automated Code Auditor Package (Section 2.4.1.5) aids in the objective measurement of the degree of adherence of the source code to the standards. Because test specifications are most effective if based on design requirements rather than the software as coded, they are developed before the corresponding code is written. In this way, errors are surfaced early when they are less costly to correct.

Structured programming is a style of programming in which the structure of a module (that is, the interrelationships of its parts) is made as clear as possible by using a restricted number of control logic structures. An important characteristic of a module written in this style is that it can be read in sequence, from top to bottom, with little or no skipping around through the code, which tends to be typical of other programming styles. This top-to-bottom characteristic extends the concept of top-down design to the actual code of each module. A module written according to structured programming principles not only has a structure but also clearly exhibits that structure.

Independent testing is a concept that maximizes objectivity of testing. This objectivity minimizes bias often found when the person who coded the software also tests the software. Thus, independent testing increases the assurance that the software satisfies the intended requirements of the System.

#### 5.3.1.2 Quality Assurance Management Plan

CSC's overall quality assurance management plan is based on the premise, which has been repeatedly demonstrated to be true, that maximizing the objectivity of the

personnel who review products significantly enhances the quality of those products. This objectivity increases the probability of the development of a reliable and acceptable System.

The key element of the plan affecting quality assurance is the establishment of a Product Assurance Technical Area independent of the Technology Analysis and Design and Software Technical Areas. The Product Assurance Technical Area defines and audits the enforcement of design, programming, testing, and documentation standards; independently reviews software designs; and independently defines and conducts integration and acceptance tests. The Product Assurance Technical Area also conducts staff training to ensure that all members of the Project are aware of, and understand, the quality assurance methodology to be employed in System design, implementation, and testing.

Another key element of CSC's quality assurance management plan is the use of structured "walk-throughs" of design, test specifications, and code. The principal participants in design and test "walk-throughs" are Project staff members from technical areas not involved in the design being reviewed. Structured "walk-throughs" are also employed by CSC at review meetings with the Government. The principal participants in code and test "walk-throughs" are programmers not involved with the code being reviewed.

#### 5.3.1.3 Quality Assurance Design Methodology

CSC's approach for designing the System is based on the concept of top-down development at all design levels, from the System level down to, and including, the module level. HIPO and data flow diagrams are used as principal design presentation methods at the subsystem, package, and subpackage levels. Module designs, on the other hand, are expressed using a design language called Program Design Language (sometimes called pseudocode in the literature), which is based upon the structured programming control logic structures described in the response to



critical issue 9 in Section III. 1.4.9 of the Specification Revisions, Initial Development Plan, and Design Analysis Report, called briefly the Task I Interim Technical Report. Specifically, CSC plans to present module designs in terms of the actual module prologs to be included in the module source decks. Structured design "walk-throughs" will be used to review all designs at all design levels.

#### 5.3.1.4 Quality Assurance Implementation Methodology

Immediately after contract award, CSC plans, as indicated in Section 5.2.1, to review the set of programming standards supplied by the Government for the System: Programming Standards for the Second-Generation Comprehensive Helicopter Analysis System.<sup>35</sup> These standards are based on the work done in Task I of this contract. Also to be reviewed is the Nomenclature of the Second-Generation Comprehensive Helicopter Analysis System,<sup>36</sup> which contains definitions of all the mathematical symbols to be used for the System.

For each module, conformance to the contract-specified programming and nomenclature standards is assessed, and nonapproved deviations are corrected before testing on the computer commences. This is accomplished using the Automated Code Auditor Package of the Support Complex and by manual inspection. The delivery of a module to the Government is accompanied by a certification of the module's conformance to the defined standards.

Before coding begins on any module, a module test specification is generated. This early generation of module test material minimizes the subjectivity that would have been introduced had the module test specification been written based on the code. When the programmer completes the module code, a code and test specification "walk-through" is conducted to ensure that the code correlates with the approved design, that the code conforms to the approved programming standards, and that the planned tests force the execution of all decision paths and includes test values critical to the functioning of the module.

#### 5.3.1.5 Quality Assurance Testing Methodology

CSC plans to apply the concepts of top-down development to testing as well as design and coding. This approach not only ensures early and exhaustive testing of the critical nucleus modules of both the Executive Component and the Technology Component, but also minimizes the amount of special test software needed, thus reducing costs. The scaffolding characteristic of top-down development automatically provides a test environment for all but general-purpose (usually mathematically oriented) modules.

The approach planned for System testing is presented in Section 5.3.2 of this report. There are four levels of testing: module, CPCI, integration, and acceptance. All decision paths, module interfaces, and CPCI interfaces are tested. Integration and acceptance testing are independently specified and conducted by the Product Assurance Technical Area. Test coverage in terms of decision paths, module interfaces, and CPCI interfaces is objectively monitored by an automated decision path monitoring test tool, the Decision Path Monitor Package. Tests to identify the central processing unit (CPU) timing characteristics and computer resource requirements for each software element are conducted during CPCI, integration, and acceptance testing. Tests for accuracy are conducted at all testing levels. Tests to demonstrate the adequacy of the analysis upon which the System design is based are conducted during integration and acceptance testing. At all testing levels, test specifications, descriptions, procedures, and reports are provided to the Government as a means of monitoring the progress and quality of System development.

#### 5.3.2 Testing Plan

This section defines the overall testing objectives for the Second Generation Comprehensive Helicopter Analysis System and the general approaches selected to accomplish these objectives. Section 5.3.2.1 defines testing objectives and concepts; Section 5.3.2.2 describes the management plan to ensure that these

objectives are met; Section 5.3.2.3 specifies the documentation necessary to evaluate the validity and progress of the testing efforts; Section 5.3.2.4 describes the automated tools to be used to support testing; Section 5.3.2.5 defines the procedures involved in resolving problems encountered during testing; and Section 5.3.2.6 describes the software library control methodology. This System Testing Plan satisfies the testing requirements defined in Section 4 (Quality Assurance Provisions) of the Baseline Type A System Specification and conforms to the Software Testing Guidance in Chapter 3 of Reference 37.

#### 5.3.2.1 Testing Objectives and Concepts

The principal objectives of System testing are threefold: to verify the accuracy of the code, identify the central processing unit (CPU) time and other computer resource utilization characteristics of the System (e.g., I/O time), and verify the adequacy of the analysis upon which the System's design is based. Four levels of testing are employed to satisfy these objectives: module testing, CPCI testing, integration testing, and acceptance testing. These levels proceed from the most detailed, module testing, to the most comprehensive, acceptance testing. The characteristics of the tests at each level are defined in terms of both the overall objectives of the testing and the orientation of the test (i.e., the basis for the assessment of success or failure). Table 5 depicts the characteristics of the tests at each testing level, and Figure 38 illustrates the relationship of each testing level to the software hierarchy.

Accuracy of coding is verified at all levels of testing. The CPU and other computer resource utilization characteristics are identified during CPCI, integration, and acceptance testing. The adequacy of the analysis upon which the System design is based is verified during CPCI, integration, and acceptance testing.

---

<sup>37</sup> RESEARCH AND DEVELOPMENT SOFTWARE ACQUISITION - A GUIDE FOR THE MATERIEL DEVELOPER, AMCP 70-4, Army Materiel Command, Headquarters U.S. Army Materiel Command, September 1974.



Table 5. System Testing Characteristics

CHARACTERISTIC TEST LEVEL	TESTING OBJECTIVE			SPECIFICATION ORIENTATION		
	ACCURACY OF CODING	TIMING AND RESOURCES	ADEQUACY OF ANALYSIS	TYPE A	TYPE B5	TYPE C5
MODULE	•					•
CPCI	•	•	•		•	•
INTEGRATION	•	•	•	•	•	
ACCEPTANCE	•	•	•	•	•	

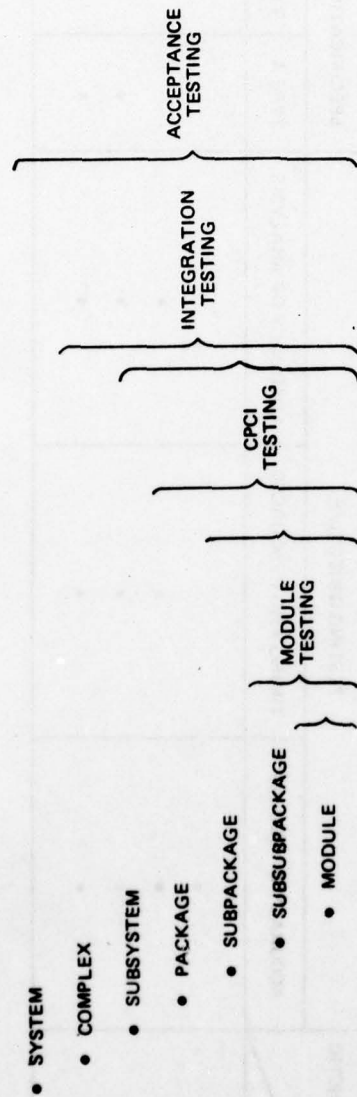


Figure 38. The Relationship of System Testing to the Software System Hierarchy

The orientation of the tests varies at each testing level. Module tests are based upon the requirements defined in the Type C5 Product Specifications; CPCI tests are based upon the requirements defined in the Type B5 Development Specifications, supplemented by the Type C5 Product Specifications; and integration and acceptance tests are based upon the requirements defined in the Type B5 Development Specifications, supplemented by the Type A System Specification. The orientation of the tests defines the basis of both test design and test evaluation.

The four levels of testing to be employed not only ensure early detection of problems but also provide an orderly methodology for mapping the overall objectives of the testing and the System specifications into the System testing effort.

#### 5.3.2.2 Test Management

Setting objectives and defining concepts for testing do not ensure that the objectives will be met and the concepts realized. Effective management of the testing is essential to meeting the defined objectives. Table 6 defines the planned extent of Government Project Office involvement in monitoring the progress of System testing. It also defines the interrelationships among the Project technical areas in the testing effort. CSC's approach to managing System testing efforts is to identify well-defined organizational responsibilities, ensure total Project involvement in testing, provide sufficient checkpoints to the Government for evaluating the efficacy and progress of System development, and apply detailed plans for the time-phased integration of development and testing efforts.

The three line organizations of the proposed Project organization (Section 5.1) are the Technology Analysis and Design Technical Area, the Software Technical Area, and the Project Assurance Technical Area. The Technology Analysis and Design Technical Area provides direct helicopter analysis support to the other two areas during all levels of testing. The support of this technical area is most critical



Table 6. Involvement of the Government Project Office and Project Technical Areas in System Testing

ORGANIZATION TEST EFFORT	GOVERNMENT PROJECT OFFICE	TECHNOLOGY ANALYSIS AND DESIGN TECHNICAL AREA	SOFTWARE TECHNICAL AREA	PRODUCT ASSURANCE TECHNICAL AREA
MODULE TEST SPECIFICATIONS	INFORMAL REVIEW	SUPPORT	RESPONSIBILITY	INFORMAL REVIEW
CONDUCT MODULE TESTS		SUPPORT	RESPONSIBILITY	SUPPORT, MONITOR
MODULE TEST REPORTS	INFORMAL REVIEW	SUPPORT	RESPONSIBILITY	INFORMAL REVIEW
CPCI TEST SPECIFICATIONS	INFORMAL REVIEW	SUPPORT	RESPONSIBILITY	FORMAL REVIEW
CONDUCT CPCI TESTS		SUPPORT	RESPONSIBILITY	SUPPORT, MONITOR
CPCI TEST REPORTS	INFORMAL REVIEW	SUPPORT	RESPONSIBILITY	FORMAL REVIEW
INTEGRATION TEST PLAN	REVIEW AND APPROVE	SUPPORT	SUPPORT	RESPONSIBILITY
INTEGRATION TEST SPECIFICATIONS	INFORMAL REVIEW	SUPPORT	SUPPORT	RESPONSIBILITY
CONDUCT INTEGRATION TESTS		SUPPORT	SUPPORT	RESPONSIBILITY
INTEGRATION TEST REPORTS	INFORMAL REVIEW	SUPPORT		RESPONSIBILITY
INTEGRATION TEST ANALYSIS REPORTS	FORMAL REVIEW	SUPPORT		RESPONSIBILITY
ACCEPTANCE TEST PLAN	REVIEW AND APPROVE	SUPPORT	SUPPORT	RESPONSIBILITY
ACCEPTANCE TEST SPECIFICATIONS	REVIEW AND APPROVE	SUPPORT	SUPPORT	RESPONSIBILITY
CONDUCT ACCEPTANCE TESTS	MONITOR	SUPPORT	SUPPORT	RESPONSIBILITY
ACCEPTANCE TEST REPORTS	REVIEW AND APPROVE	SUPPORT	SUPPORT	RESPONSIBILITY
ACCEPTANCE TESTING ANALYSIS REPORT	REVIEW AND APPROVE	SUPPORT		RESPONSIBILITY

during integration and acceptance testing because, during these two phases, the adequacy of the analysis upon which the System design is based is evaluated.

The Software Technical Area is directly responsible for all module and CPCI testing. However, the Product Assurance Technical Area reviews and monitors the module and CPCI testing efforts to ensure that sufficient tests are planned and that the tests accomplish their goals. Test specifications and reports generated during module and CPCI testing will be made available for informal review by the Government, upon request, as a means of verifying the progress of System development.

The Product Assurance Technical Area is directly responsible for integration and acceptance testing. The independence of this technical area from the other two provides the objectivity necessary to demonstrate that the System performs in accordance with the Government-approved specifications. In addition to the informal review and review/approval guidelines during integration and acceptance testing suggested by AMCP 70-4,<sup>37</sup> it is recommended that the Government monitor and participate in the actual performance of the approved acceptance tests.

Figure 39 illustrates the time-phased integration of development and testing efforts planned by CSC for the System. Preparations for each testing level represent parallel activities, thus maximizing the level of objectivity inherent at each level of testing and minimizing the effects of serialization. The time-phased integration of development and testing efforts, when combined with the Project organizational involvement in System testing planned by CSC, ensures both a reliable System testing schedule and reliable software product.

#### 5.3.2.3 Test Documentation

Table 7 shows the test documentation which is made available to the Government during all phases of System testing. Test documentation represents the principal means whereby the Government and the Development Phase Contractor can verify the progress of System development, determine if there are potential difficulties

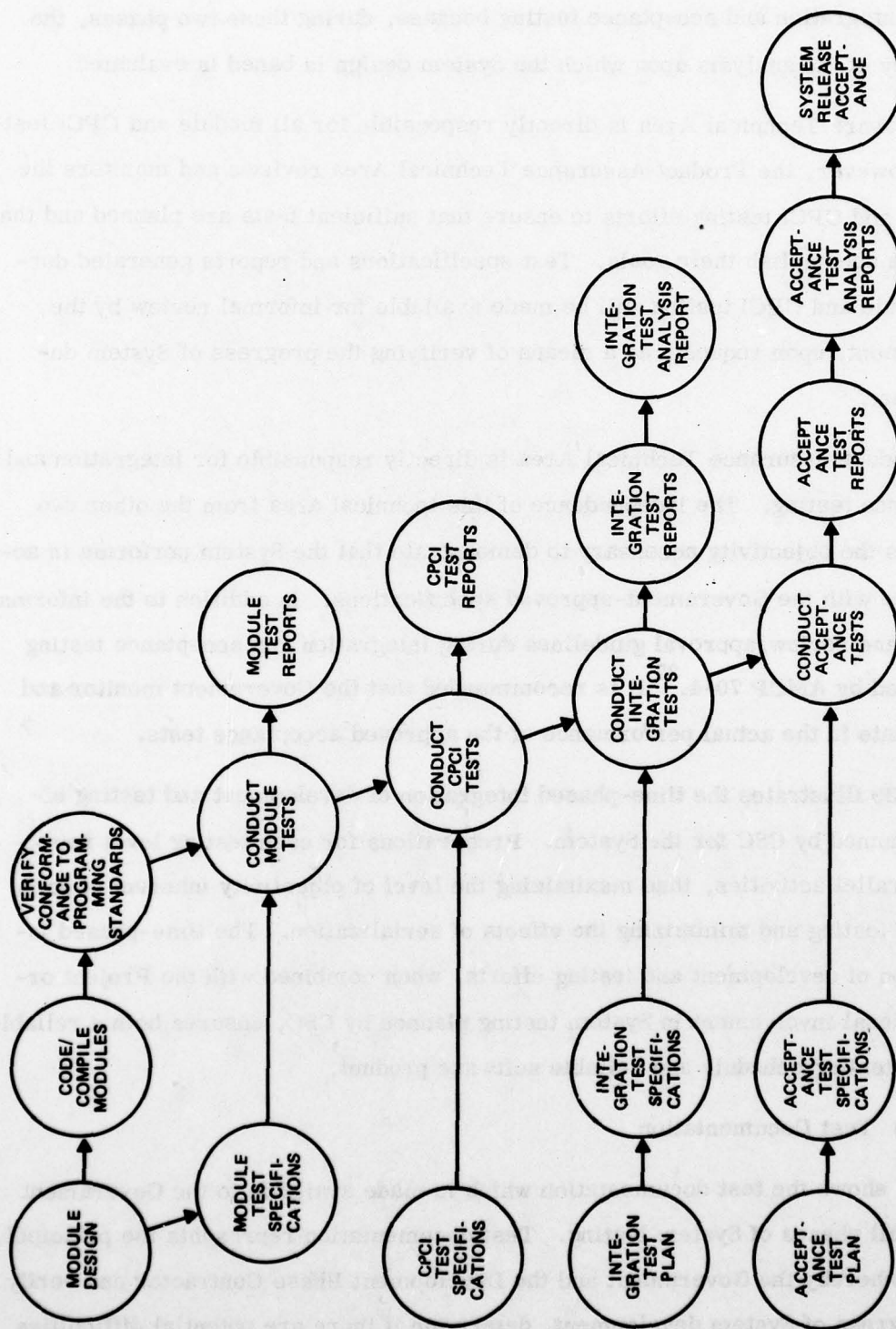


Figure 39. Development and Testing Dependencies



Table 7. Test Documentation Availability

TEST DOCUMENTATION	TESTING LEVEL	MODULE TESTING	CPCI TESTING	INTEGRATION TESTING	ACCEPTANCE TESTING
TEST PLAN					
TEST SPECIFICATIONS		INFORMAL REVIEW	INFORMAL REVIEW	REVIEW AND APPROVE	REVIEW AND APPROVE
TEST REPORTS		INFORMAL REVIEW	INFORMAL REVIEW	INFORMAL REVIEW	REVIEW AND APPROVE
TEST ANALYSIS REPORTS				FORMAL REVIEW	REVIEW AND APPROVE

or trouble areas that could affect subsequent testing or developmental efforts, and qualify the acceptability of the System.

#### 5.3.2.4 Test Tools

In the past, tools for testing software were synonymous with simulation tools. However, of at least equal importance in today's software development environment are automated tools that objectively measure the scope of the testing and objectively verify the conformance of software to defined programming standards. Table 8 shows the planned use of these two tools throughout the testing effort. Before module testing begins, the Automated Code Auditor Package, augmented by manual inspection, is used to determine each module's conformance to programming standards. All nonapproved deviations are corrected before module testing begins. The same process is independently performed by the Product Assurance Technical Area before integration testing begins.

The Decision Path Monitor Package is used during module testing to ensure that every decision path (a combination of a branch point and one of its dependent processing paths) has been tested. During CPCI testing, the Decision Path Monitor is used to ensure that every module invocation within each CPCI has been tested. During integration testing, the Decision Path Monitor is used to verify that all interface points between CPCIs have been tested. This use of the Decision Path Monitor for module, CPCI, and integration testing provides an objective method for measuring the ultimate confidence level achieved in the operational reliability of the System.

#### 5.3.2.5 Resolution of Errors

Errors encountered during testing and the costs attributable to their resolution contribute more to the unpredictability of testing schedules than any other element of System development. The software development methodology of top-down structured design and implementation, combined with rigorous enforcement of

Table 8. Relationship of Testing Level to Test Tools

QUALITY ASSURANCE TESTING LEVEL	QUALITY ASSURANCE SUPPORT TOOL	AUTOMATED CODE AUDITOR	DECISION PATH MONITOR
	MODULE TESTING		
	CPCI TESTING	•	•
	INTEGRATION TESTING	•	•
	ACCEPTANCE TESTING		•



design and programming standards, represents CSC's approach to minimizing the number of problems encountered and the cost of resolving them. However, this does not eliminate the need to define a methodology for problem resolution.

The most important element of this problem resolution methodology is explicitly defining the organizational involvements in problem identification, problem resolution, resolution verification, and resolution integration into the System. Table 9 shows the planned organizational involvement in each problem resolution effort at each stage of testing. During module and CPCI testing, problem detection and resolution are primarily the responsibility of the organizations involved in the testing, namely the Technology Analysis and Design Technical Area and the Software Technical Area. No direct involvement, beyond consulting support, by the Product Assurance Technical Area is envisioned.

However, during integration and acceptance testing, the Product Assurance Technical Area assumes a major role because of its overall responsibility for integration and acceptance testing. Product Assurance Technical Area personnel will be the principal initiators of integration and acceptance testing problem reports and will be responsible for integrating the resolutions into the System. The analysis and resolution of the individual problems by the Technology Analysis and Design and the Software Technical Areas are also monitored by the Product Assurance Technical Area to ensure that the real problem, rather than just its symptoms, is addressed. This monitoring of the analysis and resolution efforts provides the information needed to plan for the retesting necessary when the source code and other products of the resolution are delivered to the Product Assurance Technical Area for integration into the System configuration. This monitoring also provides the information needed to keep the Government abreast of the continuing progress of the testing and of potential trouble areas that could affect subsequent testing or developmental efforts.

Table 9. Project Technical Areas Involved in Resolving Problems

WHEN DETECTED EFFORTS	MODULE TESTING	PACKAGE TESTING	INTEGRATION TESTING	ACCEPTANCE TESTING
PROBLEM DETECTION	SOFTWARE	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	PRODUCT ASSURANCE	PRODUCT ASSURANCE
PROBLEM ANALYSIS	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN
SOLUTION IDENTIFICATION	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN
SOLUTION IMPLEMENTATION	SOFTWARE	SOFTWARE	SOFTWARE	SOFTWARE
SOLUTION TEST	SOFTWARE	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN
RETEST IDENTIFICATION	SOFTWARE	SOFTWARE/TECHNOLOGY ANALYSIS AND DESIGN	PRODUCT ASSURANCE/ TECHNOLOGY ANALYSIS AND DESIGN	PRODUCT ASSURANCE
SOLUTION INTEGRATION	SOFTWARE	SOFTWARE	PRODUCT ASSURANCE	PRODUCT ASSURANCE

AD-A063 921

COMPUTER SCIENCES CORP SILVER SPRING MD SYSTEM SCIEN--ETC F/G 1/3  
THE PREDESIGN PHASE OF THE SECOND-GENERATION COMPREHENSIVE HELI--ETC(U)  
OCT 78

UNCLASSIFIED

USARTL-TR-78-41

NL

4 OF 5  
AD  
A083921





#### 5.3.2.6 Software Library Controls

To ensure a responsive environment for testing, effective control of software libraries must be provided. CSC plans to establish five levels of software libraries: operational, acceptance, integration, CPCI, and module. The operational libraries reflect the source, object, and executable code of System releases accepted by the Government for operational use. The acceptance libraries reflect the differences between the current operational release of the System and the release currently undergoing acceptance testing. The integration libraries reflect the differences between the version of the System undergoing acceptance testing and the version undergoing CPCI testing. The CPCI libraries reflect the differences between the version of the System undergoing integration testing and the CPCIs being tested in the Software Technical Area. The module libraries will reflect the modules not yet ready for inclusion in the CPCI libraries for CPCI testing.

All software libraries are accessible to all Project-related personnel using the host development computers. This includes personnel from the Development Phase Contractor, integrated team member subcontractor, CPCI subcontractors, the Government, the Government/Industry Working Group, and the Technical Advisory Group. Modifications to the operational libraries require advance Government approval. However, the operational, acceptance, and integration libraries must be protected from unplanned modification. Specifically, these three libraries are updateable only by specifically identified personnel within the Product Assurance Technical Area. The CPCI and module libraries are controlled exclusively by the Software Technical Area because these libraries reflect the testing under their responsibility.

This proven software library structure minimizes the use of valuable peripheral disk storage, because each set of libraries reflects only differences from the library higher in the hierarchy of libraries. This structure also automatically provides a simple method for identifying differences between succeeding versions

of the System. Lastly, the separation of responsibility in terms of library content places the responsibility in the technical area that is actually responsible for the version of the System reflected in the libraries.

### 5.3.3 Documentation Plan

CSC's approach to the preparation and maintenance of System documentation ensures that documentation is prepared as development progresses, minimizes documentation content redundancy (and hence costs), automatically results in the documentation needed for future System maintenance without costly reformatting, and is compatible with the software engineering concepts of top-down structured design, implementation, and testing.

Based on an analysis of the documentation requirements of MIL-STD-490<sup>38</sup> (supplemented by MIL-STD-483<sup>39</sup>) and DoD 4120.17-M<sup>40</sup> for computer software systems, a documentation plan has been generated that integrates these two different sets of documentation requirements into one integrated and nonredundant set of documents consisting of the following seven documents:

1. Type B5 Computer Program Development Specification
2. Type C5 Computer Program Product Specification
3. Data Base Specification
4. User's Manual
5. System Maintenance Manual
6. Test and Implementation Plan
7. Test Analysis Report

---

<sup>38</sup> SPECIFICATION PRACTICES, MIL-STD-490, U.S. Department of Defense, October 1968.

<sup>39</sup> CONFIGURATION MANAGEMENT PRACTICES FOR SYSTEMS, EQUIPMENT, MUNITIONS, AND COMPUTER PROGRAMS, MIL-STD-483, U.S. Department of Defense, December 1970.

<sup>40</sup> AUTOMATED DATA SYSTEM DOCUMENTATION STANDARDS MANUAL, DoD Manual 4120.17-M, U.S. Department of Defense, December 1972.



Subsequent subsections of Section 5.3.3 present the outlines planned for these seven documents and describe the recommended outline modifications to MIL-STD-490<sup>38</sup> and DoD Manual 4120.17-M<sup>40</sup> needed to minimize redundancy and to allow a smooth transition from development documentation to maintenance documentation.

The most important document to be produced is the Type B5 Computer Program Development Specification because it formally establishes all the requirements that a CPCI is to satisfy. The Type C5 Computer Program Product Specification, supplemented by the Data Base Specification, actually represents the Configuration Item accepted by the Government in the sense that it is a one-to-one reflection of the software elements in the delivered Configuration Item and is therefore the prime instrument against which future software modifications are measured. The User's Manual and System Maintenance Manual are primarily aimed at providing System users and maintenance programmers, respectively, with the information needed to use and maintain the System. The Test and Implementation Plan and the Test Analysis Report provide the information needed by the Government to determine whether or not the delivered CPCI does in fact satisfy the requirements defined in the Type B5 Computer Program Development Specification. In addition, these same two documents provide baselines for verifying the impact of future modifications, and hence are also critical to the Maintenance Phase.

As mentioned in Section 5.3.1, Quality Assurance Plan, CSC plans to use four modern software design presentation concepts: HIPO (Hierarchy Plus Input, Process, Output) diagrams, data flow diagrams to illustrate the relationship between processes, module design specifications embedded in source decks, and a structured Program Design Language to describe module flow in lieu of module flow diagrams. HIPO and data flow diagrams are included in the Type B5 Computer Program Development Specification and Type C5 Computer Program Product Specification to define all functional requirements down to the module level and to illustrate the relationships among the functional requirements. All module design



specifications and flow descriptions (that are presented in Program Design Language) are included as comment cards in a module prolog in each module source deck. A listing of the module prolog will be included in the Type C5 Computer Program Product Specification as the method of presenting the functional description of the module.

CSC plans to maintain a single set of the seven documents mentioned above and described in more detail below. Because multiple CPCIs will be developed over the lifetime of the System, multiple sets of technical documentation could result. However, when a CPI is integrated into the System, the CPI (software and its associated documentation) ceases to be an independent entity. Rather, the functional characteristics of the System to date will have changed to reflect the integrated CPI. All maintenance and modification efforts subsequent to the integration of a CPI must therefore be based not on a System of independent CPCIs but rather on a System with a set of capabilities and software reflecting all CPCIs integrated into the System to date. If future maintenance efforts were based on multiple sets of documents, a severe schedule and cost impact would result. What is needed is a single integrated set of baseline technical documents reflecting all integrated CPCIs. It is this requirement that CSC plans to satisfy by maintaining a single set of the seven documents mentioned earlier.

It is recognized that, for each CPI provided by the Government or by a CPI subcontractor, a separate Type B5 Computer Program Development Specification and Type C5 Computer Program Product Specification are required. As each of these documents is accepted by the Government, it is integrated into the Baseline Type B5 Computer Program Development Specification document and the Baseline Type C5 Computer Program Product Specification document. Therefore, these baseline specifications will, at the conclusion of the Development Phase, reflect the specifications of the complete Second Level Release of the System. All updates to the Data Base Specification, User's Manual, and System Maintenance Manual provided by the Government and by CPI subcontractors will be integrated by the

Development Phase Contractor into the Baseline Data Base Specification, the Baseline User's Manual, and the Baseline System Maintenance Manual. In addition, the test specifications, descriptions, procedures, and analysis reports supplied in the Test and Implementation Plan and Test Analysis Report for each CPCI by the Government or by a CPCI subcontractor is integrated into the Baseline Test and Implementation Plan and the Baseline Test Analysis Report. To minimize the costs associated with this documentation integration effort, the outlines and standards for the baseline documents will be the same as those identified below for individual CPCIs. This documentation integration effort therefore allows a smooth transition from development to maintenance.

#### 5.3.3.1 Type B5 Computer Program Development Specification

The purpose of the Type B5 Computer Program Development Specification is to formally establish all the requirements for performance, design, test, and qualification of a Second Generation Comprehensive Helicopter Analysis System CPCI. This specification formally establishes all the requirements that the CPCI is to satisfy, i.e., it defines CPCI acceptance/rejection criteria. Following Government review and approval, any change and/or deviation from the requirements specified in the Type B5 Computer Program Development Specification are formally approved by the Government through an Engineering Change Proposal.

No deviations from the outline of the Type B5 Computer Program Development Specification, as presented in MIL-STD-490<sup>38</sup> and MIL-STD-483,<sup>39</sup> are planned. However, several additions are planned. Section 4.2, Test Requirements, will be expanded to include (a) module and CPCI testing (Internal developer testing) requirements (this will be done in Sections 4.2.1 and 4.2.2, respectively) and (b) integration testing (Preliminary Qualification Testing) requirements (this will be done in Section 4.2.3). In addition, six appendixes are specified. Appendixes I and II (Sections 10 and 20) will specify deviations to Section 3 for the Second Level and First Level Release capabilities, respectively. Appendix III (Section 30) will

include definitions of terms used elsewhere within the Computer Program Development Specification. Appendix IV (Section 40) will contain mathematical derivations referenced elsewhere within the Computer Program Development Specification. Appendixes V and VI (Sections 50 and 60) will define the programming and design standards to be followed by the CPCI contractor in designing and implementing the CPCI software. Programming standards will conform to Programming Standards for the Second Generation Comprehensive Helicopter Analysis System.<sup>35</sup> Appendixes III to VI are especially important to the success of the Second Generation Comprehensive Helicopter Analysis System because the System will not be developed as a single CPCI; rather, it will evolve from multiple CPCIs. Consistency is therefore of paramount importance to the Maintenance Phase. The primary function of Appendixes III to VI is to ensure consistency in terminology, mathematical derivations, programming, and design for the system. Without such consistency, maintenance could become a costly and protracted effort.

## TYPE B5 COMPUTER PROGRAM DEVELOPMENT SPECIFICATION

### Table of Contents

SECTION	1.	SCOPE
	1.1	Identification
	1.2	Functional Summary
SECTION	2.	APPLICABLE DOCUMENTS
	2.1	Government Documents
	2.2	Non-Government Documents
SECTION	3.	REQUIREMENTS
	3.1	Computer Program Definition
	3.1.1	Interface Requirements
	3.1.1.1	Interface Block Diagram
	3.1.1.2	Detailed Interface Definition



**SECTION 3 (Cont'd)**

**3.2 Detailed Functional Requirements**

**3.2.1 Function i**

**3.2.1.1 Inputs**

**3.2.1.2 Processing**

**3.2.1.3 Outputs**

**one set for each function**

**3.2.n\*. Special Requirements**

**3.2.n\*.1 Human Performance**

**3.2.n\*.2 Government-Furnished Property List**

**3.3 Adaptation**

**3.3.1 General Environment**

**3.3.2 System Parameters**

**3.3.3 System Capacities**

**SECTION 4. QUALITY ASSURANCE PROVISIONS**

**4.1 Introduction**

**4.2 Test Requirements**

**4.2.1 Module Testing (new)**

**4.2.2 CPCI Testing (new)**

**4.2.3 Integration Testing (new)**

**4.3 Acceptance Test Requirements**

**SECTION 5. PREPARATION FOR DELIVERY**

**SECTION 6. NOTES**

**SECTION 10. APPENDIX I, SECOND LEVEL CAPABILITY (NEW)**

**SECTION 20. APPENDIX II, FIRST LEVEL CAPABILITY (NEW)**

**SECTION 30. APPENDIX III, DEFINITIONS OF TERMS (NEW)**

**SECTION 40. APPENDIX IV, MATHEMATICAL DERIVATIONS (NEW)**

**Note: n\* = The next sequential number following the number assigned to the last function**

SECTION 50. APPENDIX V, PROGRAMMING STANDARDS (NEW)

SECTION 60. APPENDIX VI, DESIGN STANDARDS (NEW)

#### 5.3.3.2 Type C5 Computer Program Product Specification

The purpose of the Type C5 Computer Program Product Specification is to completely specify the CPCI to be formally accepted by the Government. The criticality of a Type C5 Computer Program Product Specification is best appreciated by recognizing its sensitive role in any follow-on modification and maintenance of the CPCI. For a hardware Configuration Item, the Type C5 Product Specification simply describes the Configuration Item. For a software Configuration Item, however, the Type C5 Product Specification actually is the Configuration Item in the sense that it is a one-to-one reflection of the software elements in the delivered Configuration Item and is the prime instrument against which future software modifications are measured.

Two changes, and one addition to, the Type C5 Computer Program Product Specification, as presented in MIL-STD-490<sup>38</sup> and MIL-STD-483,<sup>39</sup> are planned. All the information normally presented in Section 3.3 (Storage Allocation) is presented in the Data Base Specification instead (see Section 5.3.3.3 of this report). Therefore, for completeness, the Data Base Specification is included by reference in place of the information normally included in Section 3.3 of the Type C5 Computer Program Product Specification. All listings of the CPCI modules are included as a separate appendix (Section 20) rather than in Section 3.2. This will make the Type C5 Computer Program Product Specification easier to use as a reference document for future System modifications. The planned addition to the Type C5 Computer Program Product Specification is a special appendix (Section 10) for detailed mathematical derivations upon which the design of each Computer Program Component in Section 3.2 is based.

## TYPE C5 COMPUTER PROGRAM PRODUCT SPECIFICATION

### Table of Contents

<b>SECTION</b>	<b>1.</b>	<b>SCOPE</b>	
<b>SECTION</b>	<b>2.</b>	<b>APPLICABLE DOCUMENTS</b>	
	2.1	Government Documents	
	2.2	Non-Government Documents	
<b>SECTION</b>	<b>3.</b>	<b>REQUIREMENTS</b>	
	3.1	Functional Allocation Description	
	3.2	Functional Description	
	3.2.1	Computer Program Component 1	} One set for each Computer Program Com- ponent
	3.2.1.1	Description	
	3.2.1.2	Flow Chart	
	3.2.1.3	Interfaces	
	3.2.1.4	Data Organization	
	3.2.1.5	Limitations	
	3.3	Storage Allocation (Include the Data Base Specification by reference)	
	3.4	Computer Program Functional Flow Diagram	
	3.4.1	Program Interrupts	
	3.4.2	Logic of Computer Program Component References	
	3.4.3	Special Control Features	
<b>SECTION</b>	<b>4.</b>	<b>QUALITY ASSURANCE</b>	
	4.1	Test Plan/Procedure Cross Reference Index	
	4.2	Other Quality Assurance Provisions	
<b>SECTION</b>	<b>5.</b>	<b>PREPARATION FOR DELIVERY</b>	
	5.1	Preservation and Packaging	
	5.2	Markings	
<b>SECTION</b>	<b>6.</b>	<b>NOTES</b>	
<b>SECTION</b>	<b>10.</b>	<b>APPENDIX I, MATHEMATICAL DERIVATIONS</b>	
<b>SECTION</b>	<b>20.</b>	<b>APPENDIX II, MODULE LISTINGS</b>	



### 5.3.3.3 Data Base Specification

The Data Base Specification describes the storage allocation and data base organization and provides the basic design data necessary for the construction of the System files, tables, dictionaries, and directories. This specification not only documents the data base information needed to support maintenance efforts but also represents the focal point of data base information for all CPCI contractors. To avoid duplication of effort, the Data Base Specification is included (by reference) in Section 3.3 of all Type C5 Computer Program Product Specifications. The only change to the format of the Data Base Specification, as defined in DoD Manual 4120.17-M,<sup>40</sup> is the replacement of the Common Data Pool with the Dictionary of Computer Variables in Section 5. The Common Data Pool concept is oriented principally to online data base management systems and hence is not applicable to the Second Generation Comprehensive Helicopter Analysis System. The Dictionary of Computer Variables provides crucial design and implementation data needed by both CPCI contractors and System maintenance personnel and thus is best located in the Data Base Specification.

#### DATA BASE SPECIFICATION

##### Table of Contents

SECTION	1.	GENERAL
	1.1	Purpose of the Data Base Specification
	1.2	Project References
SECTION	2.	DATA BASE IDENTIFICATION AND DESCRIPTION
	2.1	Data Base Identification
	2.1.1	System Using the Data Base
	2.1.2	Effective Dates
	2.1.3	Physical Description of Data Base Files
	2.2	Labeling/Tagging Conventions
	2.3	Organization of the Data Base
	2.4	Special Instructions
	2.5	Support Programs Available for Handling the Data Base

SECTION 3. DATA DEFINITIONS

- 3.1 Data Files
- 3.2 Tables
- 3.3 Items
- 3.4 Records and Entries

SECTION 4. DATA BASE STORAGE

- 4.1 Internal Storage Map for the System
- 4.2 Storage

SECTION 5. DICTIONARY OF COMPUTER VARIABLES (NEW)

5.3.3.4 User's Manual

The User's Manual provides the information needed by System users and computer operations personnel. Sections 1 and 2 of the User's Manual are directed toward general management and staff personnel who have limited need for detailed technical information concerning System utilization or operation. Sections 3 and 4 present detailed information needed by engineering users who will be using the System for performing helicopter analysis. Included in Sections 3 and 4 will be instructions about providing input to the System, about user responses to System requests for information, and about the proper use/interpretation of System outputs. Section 5 contains precise, detailed information on the operating procedures necessary to successfully initiate, run, and terminate the System. Section 5 is directed toward supervisory and operator personnel responsible for the efficient performance of a computer facility. Because the System will be operational on multiple computer families, Section 5 will also contain operational information for each Government-specified host computer family.

The only change in the format of the User's Manual as described in DoD Manual 4120.17-M<sup>40</sup> is the addition of a new section (Section 5) describing computer operation procedures. This new section includes the information described in

Section 3.1 (EDP Operating Procedures) and Section 4 (Non-Routine Operations) of the Computer Operation Manual in DoD Manual 4120.17-M.<sup>40</sup> The remaining information in the Computer Operations Manual appears to be primarily oriented to complex data entry and management systems rather than to an analysis system. It is therefore CSC's plan to eliminate the generation of a separate Computer Operations Manual. Instead, the necessary computer operations information will be included in the User's Manual.

## USER'S MANUAL

### Table of Contents

SECTION	1.	GENERAL DESCRIPTION
	1.1	Purpose of the Users Manual
	1.2	Project References
SECTION	2.	SYSTEM SUMMARY
	2.1	System Application
	2.2	System Operation
	2.3	System Configuration
	2.4	System Organization
	2.5	Performance
	2.6	Data Base
	2.7	General Description of Inputs, Processing, Outputs
SECTION	3.	STAFF FUNCTIONS RELATED TO TECHNICAL OPERATIONS
	3.1	Staff Input Requirements
	3.2	Composition Rules
	3.3	Vocabulary
	3.4	Input Formats
	3.5	Sample Inputs
	3.6	Output Requirements
	3.7	Output Formats
	3.8	Sample Outputs
	3.9	Utilization of System Outputs



**SECTION 4. FILE QUERY PROCEDURES**

- 4.1 System Query Capabilities
- 4.2 Data Base Format
- 4.3 Query Preparation
- 4.4 Control Instructions

**SECTION 5. OPERATING PROCEDURES (NEW)**

- 5.1 Computer Independent Procedures
  - 5.1.1 Equipment Configuration
  - 5.1.2 Input Materials
  - 5.1.3 Output Expected
  - 5.1.4 Procedures
    - 5.1.4.1 Setup
    - 5.1.4.2 Operation
    - 5.1.4.3 Termination
  - 5.1.5 Non-Routine Operations
- 5.1 Operating Procedures for Host Computer Family 1
  - 5.1.1 Equipment Configuration
  - 5.1.2 Input Materials
  - 5.1.3 Output Expected
  - 5.1.4 Procedures
    - 5.1.4.1 Setup
    - 5.1.4.2 Operation
    - 5.1.4.3 Termination
  - 5.1.5 Non-Routine Operations

} one set for  
each host  
computer  
family

**5.3.3.5 System Maintenance Manual**

The System Maintenance Manual describes, for personnel responsible for adding new capability and maintaining the System, the procedures necessary to effectively modify and maintain the System software. The System Maintenance Manual is intended to supplement the Type B5 Computer Program Development Specification, the Type C5 Computer Program Product Specification, and the Data Base Specification for use in the future enhancement and maintenance of the System. The format presented below for the System Maintenance Manual is modeled after the format presented in DoD Manual 4120.17-M<sup>40</sup> for the Program Maintenance Manual. The differences between the planned System Maintenance and the Program Maintenance

Manual of DoD Manual 4120.17-M<sup>40</sup> result from the elimination of all duplicate information planned for inclusion in the Type B5 Computer Program Development Specification and the Type C5 Computer Program Product Specification. Sections 1.4 (Program Environment) and 1.5 (Conventions) of the Program Maintenance Manual have been eliminated because this same information will already have been included in the Type B5 Computer Program Development Specification. Section 2 (System Description) has been eliminated because this same information will already have been presented in the Type B5 Computer Program Development Specification and Type C5 Computer Program Product Specification. Section 3 (Input/Output Descriptions) has been eliminated because this information will already have been included in the Data Base Specification.

Section 2 (Computer Independent Assembling, Loading, and Maintenance Procedures) and Section 3 (Computer Dependent Assembling, Loading, and Maintenance Procedures) of the System Maintenance Manual both reflect the outline presented in DoD Manual 4120.17-M<sup>40</sup> for Section 4 (Program Assembling, Loading, and Maintenance Procedures) of the Program Maintenance Manual, with two exceptions. First, module listings will be included in Appendix II of the Type C5 Computer Program Product Specification rather than in this document; second, a new section is added to instruct the methods developer in the steps required to add new capability to the System. The reason for two sections of Assembling, Loading, and Maintenance Procedures is a recognition that the System will be operational and must therefore be maintained on different families of host computers. Assembling, loading, and maintenance procedures will therefore vary among the host computer families.

## SYSTEM MAINTENANCE MANUAL

### Table of Contents

SECTION 1.	GENERAL DESCRIPTION	
1.1	Purpose of the System Maintenance Manual	
1.2	System Allocation	
1.3	Equipment Environment	
SECTION 2.	COMPUTER INDEPENDENT ASSEMBLING, LOADING, AND MAINTENANCE PROCEDURES	
2.1	Input/Output Requirements	
2.2	Procedures	
2.3	Verification	
2.4	Special Maintenance	
2.5	Other Special Maintenance Packages and Procedures	
2.6	Error Conditions	
2.7	Adding New Capability to the System (NEW)	
SECTION 3.	COMPUTER-DEPENDENT ASSEMBLING, LOADING, AND MAINTENANCE PROCEDURES	
3.1	Procedures for Host Computer Family 1	
3.1.1	Input/Output Requirements	} one set for each host computer family
3.1.2	Procedures	
3.1.3	Verification	
3.1.4	Special Maintenance Packages and Procedures	
3.1.5	Other Special Maintenance Procedures	
3.1.6	Error Conditions	

#### 5.3.3.6 Test and Implementation Plan

The Test and Implementation Plan is a tool for directing the integration and acceptance testing effort associated with a CPCI. It contains the schedule of events and list of materials necessary to effect delivery of the CPCI and to



conduct the orientation required for proper use of the CPCI. Specifically, this plan has four objectives:

1. To provide guidance for the management and technical effort necessary throughout testing
2. To establish a comprehensive test plan and to communicate to the Government the nature and extent of the tests deemed necessary to provide a basis for acceptance of the CPCI
3. To provide an orderly schedule of events, a specification of equipment and organizational requirements, the methodology of testing, a list of materials to be delivered, and a schedule for Government orientation
4. To provide a written record of actual test inputs used to exercise CPCI limits and critical capabilities, the instructions to permit execution of the tests by Government personnel, and the expected outputs

Sections 1, 2, and 3 will address both integration testing and acceptance testing efforts. Sections 4 and 5 will include the test descriptions and test procedures for acceptance testing. Test descriptions and test procedures for integration testing will be made available for informal review upon request of the Government as the material becomes available.

The only change in the outline of the Test and Implementation Plan as described in DoD Manual 4120.17-M<sup>40</sup> is the placement of test evaluation information. In DoD Manual 4120.17-M,<sup>40</sup> test evaluation information is presented in a separate section (Section 6) of the Test and Implementation Plan and is oriented to the total testing effort. For the Second Generation Comprehensive Helicopter Analysis System, test evaluation information is more likely to be a function of each test

than a function of the collection of all tests. Therefore, test evaluation information has been added as a requirement to each test procedure (Section 5).

## TEST AND IMPLEMENTATION PLAN

### Table of Contents

SECTION	1.	GENERAL
	1.1	Purpose of the Test and Implementation Plan
	1.2	Project References
	1.3	Deliverable Materials
	1.4	Statement of Pre-Test Activity
SECTION	2.	TEST PLAN
	2.1	CPCI Description
	2.2	Test Milestone Chart
	2.3	Environment
	2.3.1	Equipment Requirements
	2.3.2	Software
	2.3.3	Personnel
	2.4	Training Plan
	2.5	Test Materials
SECTION	3.	TEST SPECIFICATIONS
	3.1	Requirements
	3.2	CPCI Functions
	3.3	Test/Function Relationships
	3.4	Test Methods and Constraints
	3.4.1	Test Conditions
	3.4.2	Test Means of Control
	3.4.3	Extent of Test
	3.4.4	Data Recording
	3.4.5	Test Constraints

**SECTION 4. TEST DESCRIPTIONS**

4.1	Test Progression	}	one for each test
4.1	Test (Identify) Description		
4.1.1	Test Inputs		
4.1.2	Test Outputs		
4.1.3	Test Conditions		
4.1.4	CPCI Conditions		
4.1.5	Test Control		
4.1.5.1	Input Data		
4.1.5.2	Input Commands		
4.1.5.3	Output Data		
4.1.5.4	Output Notification		

**SECTION 5. TEST PROCEDURES**

5.1	Test (Identify) Description	}	one for each test
5.1.1	Test Setup		
5.1.2	Test Initialization		
5.1.3	Test Steps		
5.1.4	Test Termination		
5.1.5	Test Evaluation (new)		
5.1.5.1	Test Data Criteria (new)		
5.1.5.2	Test Data Reduction (new)		

**5.3.3.7 Test Analysis Report**

The Test Analysis Report describes the status of the CPCI after testing. This document has four objectives:

1. To document the results of integration and acceptance testing of CPCI
2. To provide a basis for allocating responsibility for deficiency correction and followup
3. To provide a basis for the preparation of a Project completion statement
4. To establish user confidence in the utilization of the CPCI



Two Test Analysis Reports will be produced for each CPCI. The first report will cover the integration testing effort. The second report will cover the acceptance testing effort. No modifications to the outline of the Test Analysis Report as described in DoD Manual 4120.17-M<sup>40</sup> are planned.

## TEST ANALYSIS REPORT

### Table of Contents

SECTION	1.	GENERAL	
	1.1	Purpose of Test Analysis Report	
	1.2	Project References	
SECTION	2.	TEST ANALYSIS	
	2.1	Test (Identify)	} one for each test
	2.1.1	Data Performance	
	2.1.2	Parameter Performance	
SECTION	3.	CPCI FUNCTION ANALYSIS	
	3.1	CPCI Function (Identify)	} one for each CPCI function
	3.1.1	Function Performance	
	3.1.2	Performance Limits	
SECTION	4.	SUMMARY AND CONCLUSIONS	
	4.1	Demonstrated Capability	
	4.2	CPCI Deficiencies	
	4.3	CPCI Refinements	

#### 5.3.4 Subordinate Technical Plans

The first two subordinate technical plans presented in this section, the Data Processing Facility Plan (Section 5.3.4.1) and the Data Management Plan (Section 5.3.4.2), are presented, by and large, intact from the Baseline Development Plan. The three remaining subordinate technical plans, the System Installation and Release Plan (Section 5.3.4.3), the Training Plan (Section 5.3.4.4), and the Maintenance Plan (Section 5.3.4.5), have been summarized from similarly titled plans in the Baseline Development Plan.

#### 5.3.4.1 Data Processing Facility Plan

This section presents the computer facilities needed to support the development of the Second Generation Comprehensive Helicopter Analysis System and the data processing facilities available through commercial vendors to support System development. Mainframe computer equipment, operating system software, programming support software, interfacing terminals, and assurance of access to CPU time are discussed.

Because of the desirability for all helicopter manufacturers and Government users to have the System on their computer, the Baseline Type A System Specification has specified that the System be developed on the following computer configurations: the IBM S/370 Models 158 and 168 under OS/VS, the IBM S/360 Model 65 under OS/MVT, and the CDC 6600 and CYBER series under NOS. To support the development of the System, two development computers are required: the Host 1 Development Computer and the Host 2 Development Computer. The IBM S/370 and S/360 computers are collectively called the Host 1 computer, and the CDC 6600 and CYBER series computers are collectively called the Host 2 computer. The Host 1 Development Computer configuration is a large-scale IBM S/370 (Model 158 or 168) operating under the Multi-Virtual Storage (MVS) system. The Host 2 Development Computer configuration is a CDC CYBER-175 using the Network Operating System (NOS). Such systems are available from several commercial vendors, who have provided pricing information and assurances that their throughput capacities will be more than ample to support the projected requirements for development of the System.

##### 5.3.4.1.1 Host 1 Development Computer Configuration

The commercially available configuration proposed for use during System development typically consists of one or more IBM S/370 Model 168 processing units of core capacity equal to or greater than 5 megabytes with all associated channeling and controllers and a comprehensive array of peripherals. Total disk capacity

well in excess of 5000 megabytes is common among the vendors. The computer operating system, the Multi-Virtual Storage System, is a highly sophisticated system that protects both the Development Phase programmer using the Time Sharing Option (TSO) and the batch program from many kinds of hardware failure; offers enhanced security provisions; and supports the standard IBM languages, utilities, and subsystems. The virtual memory storage feature makes efficient use of the actual memory, or real storage, of the CPU by keeping program instructions and data in real storage only when currently required for execution. The rest of a program remains in external page storage, ready to be "paged in" by the operating system to memory when needed. Thus, an executing program will "see" a memory capacity many times greater than the actual CPU storage.

The vendors interviewed by CSC further support their systems with extensive object program libraries, macro libraries, and test and debug libraries, as well as a large array of programming languages. The IBM S/370-168 is interactive to an extensive array of TSO and graphics types of terminals, spanning a wide range of manufacturers and models, and may also be accessed by both remote job entry and standard batch submission. A remote job station, located at CSC's Silver Spring, Maryland, facility providing remote batch and interactive capabilities, could be easily supported by all the vendors surveyed. To enhance mainframe accessibility, a dedicated, leased-line system, with 24-hour accessibility, is also a standard offering.

#### 5.3.4.1.2 Host 2 Development Computer Configuration

The commercially available computer facilities available to support development of the System for the proposed Host 2 Development Computer, the CYBER-175, typically consist of two or more CYBER-175 central processor units, operating under NOS, with real memory capacity of 262,000 60-bit words. Each CYBER-175 is also a virtual storage computer having the advantages described in Section 5.3.4.1.1, with up to 2 million 60-bit words of extended core storage.



Mainframes share a common permanent file system and extended core storage. Furthermore, the System is supported by extensive object program libraries, macro libraries, test and debug libraries, and a large assortment of programming languages.

Interactive capability is available for an extensive array of both alphanumeric and graphics types of terminals through NOS. These capabilities are also accessible by remote job entry or standard batch submission. As with the Host 1 Development Computer configuration, vendors of Host 2 Development Computer services indicated that they could easily support a remote job station at CSC's Silver Spring, Maryland, facility, including both remote batch and interactive capabilities. To enhance mainframe accessibility, a dedicated, leased-line system, with 24-hour accessibility, is also a standard offering.

In addition to the CYBER-175, access to CDC 6000 and CYBER-70 series computers is provided by the same vendors under the same operating system, i.e., NOS.

#### **5.3.4.1.3 Terminals**

Vendors for both configurations of host development computers support a wide variety of terminals. High-speed, remote-job-entry interfaces with line speeds up to 9600 baud and the accompanying line printer and card reader/punch are supported, as well as many types of teletype or CRT graphic and interactive terminals.

Upon contract award, CSC will determine the most appropriate hardware, which will be acquired by lease, for establishment of a remote batch and interactive station to be dedicated to the development of the System.

#### **5.3.4.1.4 Access to CPU and Turnaround Time**

Guaranteed turnaround time is provided by commercial vendors on a graduated cost basis. The faster the turnaround desired, the greater the cost, with

interactive access being the most expensive. Of course, interactive processing in many development applications can also be the most productive. The graduation is effected through input priority "classes" requested by the user when a job is submitted for processing. However, all of the vendors contacted expressed confidence that, because of their present capacities and planned, near-future expansions, CSC would, on the whole, experience much faster turnaround time than guaranteed by the priority class request.

#### 5.3.4.1.5 Programmer Support

The software available for programmer support through commercial service vendors is extensive. Higher level language compilers such as FORTRAN, COBOL, and PL/I are universally available, as well as the assembly languages for the respective host development computers. Services for source program configuration control, code documentation, and performance testing--all of which greatly enhance the efficiency of the programmer--are supported by the vendors contacted by CSC. Application programs and systems spanning a wide variety of subjects, such as data base management, structural analysis, and graphics, are also available. In conjunction with many kinds of terminals, the graphics software supported by vendors can produce displays (interactive or hardcopy) from the simplest x-y plot to complex three-dimensional representations.

Additionally, CSC has interviewed several qualified minority contractors who could provide card-punch services required during the development effort.

#### 5.3.4.2 Data Management Plan

To reduce risk, software and data are stored in disk files protected from accidental destruction by normal measures--passwords and periodic backup copies. There are several versions of the System available for use. One version is the last completed build. (A build is a subset of the System; see Section 5.4 for a discussion of builds.) This version of the System is maintained in a stable condition so that it is available for use by the various organizations involved (e.g., CPCI

subcontractors). Other versions of the System serve as stages toward the completion of the next build. As CPCIs are completed, they are integrated into one of these versions of the System and are tested with the remainder of the System being integrated for the next build. Procedures are applied to maintain libraries of source modules from which each version of the System is produced and to maintain the various System versions in executable form.

A large amount of test data is required during System development. Initial versions of the Master Data Base and other files against which tests are to be run are maintained in association with the input data and the results expected from the test. Sets of test data correspond to versions of the System. Associated with the last completed build in disk and card files are all tests run to verify that build, and the printed or plotted results of those tests. Each version of the System representing a stage in the development of a build has associated sets of test data, including valid data as well as invalid data. The maintenance of the various libraries and test data files, their status, and their associations is accomplished by establishing procedures for the development of test data, the integration of modules into partially completed Systems, the execution of tests, and the recording of test results.

#### 5.3.4.3 System Installation and Release Plan

The System Installation and Release Plan establishes the process to install the System on the host computers specified by the Government and to provide each facility with a complete release product set. A plan for the installation of only the First Level Release would not be complete without the inclusion of the installation of the Second Level Release or the provision for intermediate releases between them. Installation is discussed in Section 5.3.4.3.1, and release is discussed in Section 5.3.4.3.2.

##### 5.3.4.3.1 Installation

CSC recommends that the installation of the First Level Release be done by the Development Phase Contractor at each computer facility specified in the



Development Phase contract. An installation is performed by personnel from the Product Assurance Technical Area who generate the First Level Release of the System and perform on-site verification tests to ensure the System is functioning properly. It is recommended that Government personnel, who will be responsible for System maintenance after the completion of the Development Phase, be in attendance to observe the techniques required to install the System on various host computers. The reason for this recommendation is that CSC has a supplementary recommendation for the installation of the Second Level Release: the reversal of the roles of the Development Phase Contractor and Government personnel to generate and install the Second Level Release of the System. Specifically, those Government personnel responsible for the System maintenance would install the System, with the Development Phase Contractor personnel assisting in a consulting role at the computer facilities specified in the Development Phase contract. This approach provides Government personnel with an understanding of the installation process because they will have been exposed to it at First Level Release, will have witnessed the System-generation tasks, and will have participated in the discussion and resolution of problems that may develop during the event. Witnessing the First Level Release installation also assists in the preparation of the plan for the installation of the Second Level Release.

It is also recommended that one or two intermediate releases of the System be considered during the first 2 years of the Validation Phase. The purpose of an intermediate release is to provide new capability to the user community and provide corrections to the First Level Release. In addition to enhancing the System (and hence its acceptance), an intermediate release can serve as a phase-in activity for Government personnel charged with eventual responsibility for maintenance. Because the Development Phase Contractor is responsible for the installation of the First Level Release and presuming the Government installs the Second Level Release (with Development Phase Contractor help), then a gradual

shifting of the roles between the two levels provides for a smooth transfer of responsibility.

#### 5.3.4.3.2 Release

A release product set for the First Level Release and the Second Level Release of the System contains complete documentation, programmer installation instructions, tapes, cards, listings, and any other materials necessary to understand, use, operate, and maintain the System. Details on the contents of the release product set can be found in Section 4.6.2 of the Baseline Development Plan.

#### 5.3.4.4 Training Plan

Training in the System includes basic instruction for First Level Release users (both engineering users and methods developers) to introduce them to the System and advanced instruction for Second Level Release users who have prior experience with the System. CSC also strongly recommends training for programmers who will install local (onsite) corrections, aid methods developers in incorporating new modules of the Technology Component in the System, and eventually maintain, locally, the System for their company or agency. Between the First Level and Second Level Releases, it is recommended that the Government consider review training for both new and experienced users coincident with intermediate releases of the System.

The training of users and programmers should be scheduled as nearly coincident with the System installation as possible to heighten the positive impact of the System. This recommendation implies that theoretical concepts should be presented before or during installation and practical instruction should begin as soon as the System is available at a facility.

Thorough training, at the time of each release, typically takes 2 weeks of instruction. The first week includes the mathematical basis of the System and illustrative examples, and the second week is devoted to the study of computer executions.

Training coincident with an intermediate release would be for 1 week because this instruction is more specialized to review current concepts and to study new developments introduced with the intermediate release.

Two training courses are recommended: one for users and the other for programmers. The user training course provides engineering users and methods developers the knowledge of the capabilities and use of the System to allow them to obtain meaningful results from the System and to be able to add to or change software elements in the System. The programmer training course introduces the programmer to the software features and requirements of the System. The programmer's responsibilities encompass System installation, System modification, aiding the methods developer in the permanent incorporation of new Technology Component capabilities into the System, and understanding user requirements.

Outlines for user courses and programmer courses are provided in the Baseline Development Plan.

It is anticipated that the Government will play an active management role during maintenance activities. This role includes receiving from users the description of potential problems, providing the initial estimate of the resources required to incorporate additions to the System, and being directly involved in determination of what and how information is to be disseminated to users in response to their problems. To foster positive relationships with users and programmers, there must be a form of personnel assistance and consultation services available to them.

It is important to train System users in the Government's procedures for user communication of information about errors and about suggestions for incorporation of enhancements. The users must know how the maintenance cycle operates, how the maintenance activity can be used to assist them, and what support services are available. To acquaint users and programmers with the services available to them, some instruction relative to these services must be included in the



training plan and should address maintenance, consultation, and assistance services.

#### 5.3.4.5 Maintenance Plan

The discovery of errors in the System and timely correction of them are of paramount importance to preserve the integrity of the System. Equally important are the immediate assessment of the error (and its impact on the System) and the determination of an adequate correction to provide the user with a viable solution to his or her problem. Both of these latter points involve response to the user to identify critical deficiencies and to incorporate corrective action. Included with the identification of critical deficiencies is the determination of evasive action on the part of the user to avoid the deficiency.

There are several possibilities for disseminating information to users to be responsive to error conditions and corrections. Examples are bulletins distributed to all users, individual responses to users with *less critical problems*, and automated data bases intended for user access. The first two approaches are simple and straightforward forms of dissemination media but have the inherent disadvantage of the amount of distribution time required. The data base concept allows instantaneous access to problems and corrections but includes higher costs in terms of maintenance and labor. All of these possibilities could be utilized as complementary approaches. Cost feasibility and System criticality dictate the "mix."

Consultation and assistance to System users and programmers are vital functions that must be performed to ensure they obtain maximum benefit from the System. Direct personal relationships with the users and programmers enable Development Phase Contractor personnel to assess the acceptability, usability, and application of the System through personal communication and feedback. The benefits derived are that the Development Phase Contractor can reevaluate

such important efforts as training and documentation to clarify uncertainties on the part of the users and programmers.

The Maintenance Plan defines the approach and procedures to be used for identifying errors and deficiencies in the System, incorporating additional modules and CPCIs into the System, and disseminating correction information to users during the Validation Phase. The Validation Phase begins with the First Level Release of the System and ends 1 year after the completion of the Development Phase. During the Validation Phase, the System will be subjected to rigorous testing and evaluation by users from industry and Government.

The maintenance of the System includes the identification of errors and deficiencies in the System, the timely incorporation of corrections to errors and deficiencies as well as the addition of enhancements and new capabilities to the System, the test and verification of the System after the incorporation of corrections and additions, the updating of System documentation, and the generation and release of an improved System.

To keep the Government/industry user community apprised of the progress and latest developments occurring during maintenance, reports to the Government by the Development Phase Contractor are required; dissemination of information to the users by the Government is also required.

To keep the originator of change requests and all users (when required) apprised of current maintenance activities, dissemination of information is appropriate at several steps in the maintenance cycle. (The maintenance cycle is described in some detail in Section 4.8 of the Baseline Development Plan.) After the originator has submitted a request to change the System, the Government should notify the originator that the request has been received and an analysis of it is in progress.

During analysis of the problem, the initial assessment of an error with respect to its impact on the System dictates the necessity for dissemination of information

to recognize that an error exists and that a possible alternative (temporary) solution to evade the problem or to utilize a different approach to achieve a solution equal to the attempted original one is available. In the event that an apparent error does not exist or that the error does exist and is known by previous means, the originator should be informed of the determination. When an error exists (for which no evasion can be identified) that is being corrected, all users should be made aware of these conditions.

When the solution to the problem has been verified, System modifications may be disseminated for System updating at the local site. This is particularly important if a temporary solution to an error could not be identified during problem analysis. At this time, enhancements or additions can be "advertised" as ready for general use through a modification to be made at the local site.

Following the Government's acceptance of new or changed documentation, synopsis information or full text may be disseminated, depending upon the subject matter. Material related to error corrections could be outlined in brief form, whereas new capabilities or enhancements would require that full text be available to adequately derive full benefit from the addition to the System.

The use of the System on different hardware configurations introduces the problem of adequate and equal maintenance on all host computers. The effectiveness of the System requires the maintenance of both computer-dependent and computer-independent software. Maintenance of computer-independent software has some impact on the operation of the System because the acceptability of the code on each host computer must be taken into account.

Maintenance of computer-dependent software is much more inclusive and complex: overlay structure, file management, and module communications differences on each host computer require thorough integration and acceptance testing to ensure equal treatment for all capabilities of the System. System utilization and load



procedures must be evaluated and monitored to achieve maximum efficiency on all host computer families.

The Software Technical Area and the Product Assurance Technical Area are structured to include specialists for each host computer family. This approach guarantees concurrent System updates for each host computer. It also ensures expert response to System users' and programmers' questions as they relate to specialized problems.

#### 5.4 IMPLEMENTATION PLAN

This Implementation Plan establishes a preliminary time-phased plan for developing the First Level Release and the Second Level Release of the Second Generation Comprehensive Helicopter Analysis System. The time-phased plan presented in this section is based upon four considerations. First, the basic operational and software environment, upon which the overall System design is based, will be developed early in the Development Phase. Second, System capabilities will be developed incrementally and integrated incrementally into the System throughout the Development Phase. Third, proper monitoring by the Government of the development effort calls for maximum visibility. And fourth, acceptance of the System by the helicopter firms will be enhanced by participation of the helicopter firms in the System development efforts. Because of the first two considerations, the Implementation Plan has been shaped by the strategy of builds, which is a powerful, proven software implementation strategy that minimizes risk. Because of the last two considerations, each functional capability of the System has, for the purposes of the Predesign Phase, been identified as a separate Computer Program Configuration Item (CPCI). The number of CPCIs to be defined in the Development Phase will be considerably fewer than the 204 CPCIs identified during the Predesign Phase. Because of the size and complexity of the System, CSC recommends strongly that it be developed and tested incrementally in a series of builds leading

to the First Level Release capability and later to the Second Level Release capability. A build, which is a subset of the entire System, provides a demonstrable functional capability that is a subset of the total functional capability of the System. This "build-a-little, test-a-little" philosophy has several advantages over the alternative strategy of developing all of the software elements required to produce the First Level Release capability and then all the software elements required to produce the Second Level Release capability. These advantages are:

- Any major interface problems will be discovered in the testing of the first or second build when they are more easily corrected.
- The software integration effort, a source of many problems in past software development efforts, is spread more smoothly over the entire Development Phase rather than being placed near the end. Risk is thus reduced.
- A reasonably stable and well-defined partial System is available for testing following the first build.
- A part of the total System capability is demonstrated to the System's acquirer and to System users early.

To realize these advantages, the builds must be judiciously specified in terms of functional capability and content. Each build must have a demonstrable functional capability that is a subset of the total System functional capability. A given build contains all the capabilities of the previous build plus new capabilities. Capabilities upon which many parts of the System are dependent (most Executive Component capabilities fall into this category) are included at an early point in the build sequence. Upon completion of a build, it is maintained in a stable condition for use in testing the elements of future builds.

During the incremental development of the System, the Development Phase Contractor is responsible for integrating the CPCIs into builds regardless of the

organization responsible for the development of the CPCI (Development Phase Contractor, CPCI subcontractor, or Government). In each case, a preceding build, which has been verified, is available for module testing, CPCI testing, integration testing, and acceptance testing by the organization responsible for the development of a CPCI.

The Implementation Plan is composed of two subplans: the Operational Complex Implementation Plan (presented in Section 5.4.1) and the Support Complex Implementation Plan (presented in Section 5.4.2). The Operational Complex is the part of the System that the engineering user accesses to obtain predictions of performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability for a variety of rotary-wing aircraft configurations. The Support Complex is the part of the System that is used to support the development, testing, configuration control, and documentation of the total System. Each complex is composed of packages and subpackages, each of which represents a functional capability of the System. Each subpackage of the System has, for the purposes of the Predesign Phase, been identified as a separate CPCI. Packages (that have not been subdivided into subpackages) have also been identified as CPCIs. Because each functional capability of the System has been identified as a separate CPCI, not only is the Government's visibility into CSC's plan for implementation in the Development Phase maximized, but also the potential participation by helicopter firms in the development of System capabilities is maximized.

Table 10 summarizes the detailed sizing and professional labor estimates specified in each of the implementation subplans. Eight builds (builds 1-8) have been identified for the Operational Complex; four builds (builds A-D) have been identified for the Support Complex. Builds 1 to 4 and A through D represent the First Level Release on the Host 1 computer family; build 5 represents the First Level Release on the Host 2 computer family; and builds 6, 7, and 8 represent the capabilities added to the First Level Release for the Second Level Release.



Table 10. Summary of System Size and Professional Labor Estimates

BUILD	TECHNOLOGY COMPONENT		EXECUTIVE COMPONENT		OPERATIONAL COMPLEX		SUPPORT COMPLEX	
	ESTIMATED LINES OF EXECUTABLE CODE	ESTIMATED DEVELOPMENT LABOR (MAN-MONTHS)	ESTIMATED LINES OF EXECUTABLE CODE	ESTIMATED DEVELOPMENT LABOR (MAN-MONTHS)	ESTIMATED LINES OF EXECUTABLE CODE	ESTIMATED DEVELOPMENT LABOR (MAN-MONTHS)	ESTIMATED LINES OF EXECUTABLE CODE	ESTIMATED DEVELOPMENT LABOR (MAN-MONTHS)
1			11100	79.0	11100	79.0		
2	11350	68.5	6100	41.5	17450	110.0		
3	13300	90.5	7100	44.0	20400	134.5		
4	7250	45.5	2000	11.0	9250	56.5		
5			2000	15.0	2000	15.0		
6	19850	138.0	2000	11.0	21850	149.0		
7	18400	149.5	13000	98.5	31400	248.0		
8	4100	40.5	2500	15.0	6600	55.5		
A							1000	7.5
B							4500	33.0
C							8000	58.0
D							3000	20.5
TOTAL	74250	532.5 (44.4 MAN-YEARS)	45800	315.0 (26.2 MAN-YEARS)	120050	847.5 (70.6 MAN-YEARS)	16500	119.0 (9.9 MAN-YEARS)

TOTALS FOR SYSTEM: ESTIMATED LINES OF EXECUTABLE CODE = 136550; ESTIMATED DEVELOPMENT LABOR (MAN-MONTHS) = 966.5 (80.5 MAN-YEARS)

No estimates are indicated for Technology Component software in build 1 because build 1 is planned to provide the basic Executive Component capabilities needed to develop all subsequent Operational Complex capabilities. Builds 2 and 3 provide the major portion of the First Level Release helicopter analysis capabilities; builds 6 and 7 provide the major portion of the Second Level Release helicopter analysis capabilities to be added to the First Level Release. Builds 4 and 8 are intentionally planned to be relatively small builds because each represents the last build in an operational release of the System and will therefore require acceptance-level testing in addition to integration testing. In addition, problems or inconsistencies discovered in prior builds can be corrected in these two small builds with the least potential impact on the development schedule. Another special build is build 5. Because this build provides the First Level Release capabilities on the Host 2 computer family, only the Executive Component is affected. No Technology Component size and labor estimates are supplied for build 5 because no Technology Component software will be affected by this extension of the System to the Host 2 computer family.

The number of lines of executable code for the Second Level Release of the System is estimated at 136,550 and the professional labor for development of this capability is estimated at 966.5 man-months (80.5 man-years).

Each of the two implementation subplans is presented in the form of a development schedule for each complex and a sequence of build tables. The development schedules identify the major development activities required to develop the First Level and Second Level Releases, and show the time-phased plan for the development of each build in each Complex. For each build, a summary description of the operational capabilities (functions) provided by the build is presented at the beginning of each table. Each build is identified in terms of the packages or subpackages to be included in it. In addition, the packages and subpackages in each build are presented in the hierarchical framework of the System so that the functional relationships can be understood.

For each software element of the build, four factors are identified which provide information essential to planning and control. These four factors are: Estimated Lines of Code, Estimated Development Labor (Man-Months), Estimated Memory Requirements (Bytes), and Candidate CPCI Source. These four factors are presented in four columns for each build. The Estimated Lines of Code column of the build tables is an estimate of the number of executable FORTRAN statements to be included in each software element. The Estimated Development Labor column in each build table represents the estimated professional man-months needed to develop the software element. The labor estimates include all the development activities (i.e., analysis, design, code, test, documentation, training, etc.) required by the organization responsible for the development of the software element.

The Estimated Memory Requirements column of the build tables represents an estimate of the number of bytes of memory required on an IBM S/370 (or S/360) computer. These estimates represent the total memory required for both instruction storage and local data storage. While data storage was estimated as a function of capability, instruction storage estimates were based on an average of either six computer instructions (each requiring an average of four bytes) per executable FORTRAN statement for highly computational software elements or four computer instructions per executable FORTRAN statement for all other software elements.

The last column of each build table (Candidate Software Source) identifies a potential source of the software element. While it is theoretically possible that each software element could be provided by independent contractors, the resulting costs and risks associated with this approach are unacceptable. The approach selected was to identify common potential sources for groups of software elements so that development can proceed in parallel and independently. This approach minimizes the impact that one development effort can have on a parallel development effort. In addition, two assumptions affected the identification of candidate software sources: (1) few, if any, of the software elements planned for inclusion in the



First Level System Release would be Government furnished, and (2) few, if any, of the software elements planned for inclusion in the Second Level System Release would be furnished by subcontractors.

#### 5.4.1 Implementation Plan for the Operational Complex

Eight builds are planned for the Operational Complex. The first build establishes a basic operational and software environment so that subsequent development efforts can proceed in parallel. Builds 2, 3, and 4 provide capabilities required in the First Level System Release. Build 4 is the equivalent to the First Level Release capability on the Host 1 computer family, i.e., the IBM 360/370 computer family. Build 5 is the equivalent of the First Level Release capability on the Host 2 computer family, i.e., the CDC 6000/CYBER computer family. Builds 6, 7, and 8 provide the added capabilities required in the Second Level Release. Build 8 is the equivalent to the Second Level Release capability on both the Host 1 and Host 2 computer families.

Figure 40 presents the planned schedule for developing the First and Second Level Releases of the Operational Complex. The schedule indicates that the First Level Release will be available for Government acceptance testing midway through the four-year Development Phase. It is expected that the First Level Release will be ready for installation on Government-specified IBM S/370 and S/360 computers throughout the helicopter analysis community 3 months after initiation of Government acceptance testing. The Second Level Release of the Operational Complex will be available for Government acceptance testing 3 months before the end of the Development Phase. The Second Level Release will be ready for installation on Government-specified IBM 360/370 computers and CDC 6000/CYBER computers throughout the helicopter analysis community by the end of the fourth year in the Development Phase.

The 15 milestone events presented in Figure 40 provide the visibility needed by the Government to assess the progress of the Development Phase, to verify that

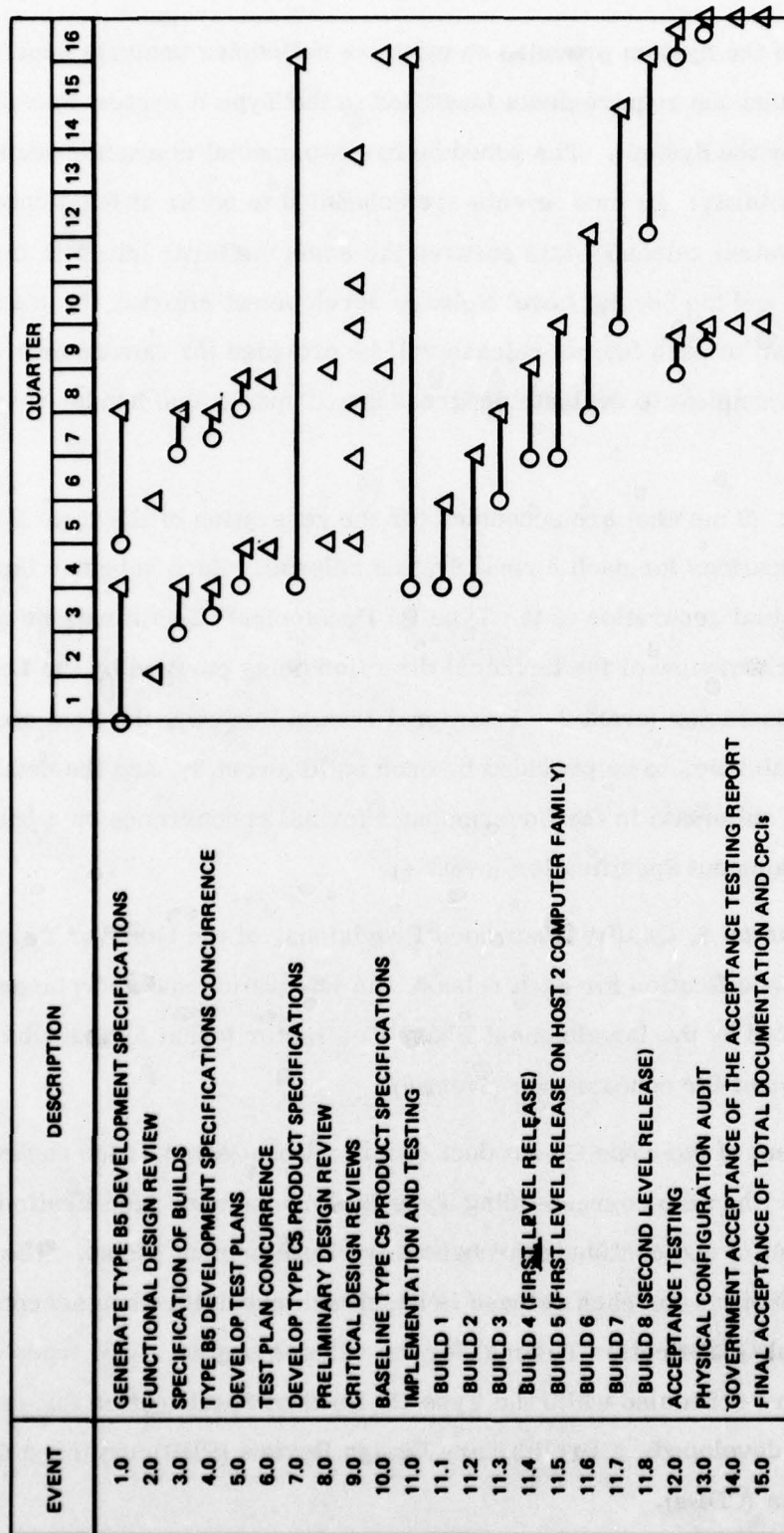


Figure 40. Milestone Schedule for Development of the Operational Complex

each release of the System provides an effective helicopter analysis capability, and to ensure that the requirements identified in the Type A System Specification are satisfied by the System. The schedule has two special characteristics which ensure this visibility: (1) most events are scheduled to occur at least once for each formal System release - this ensures the same visibility into both the First Level Release and the Second Level Release development efforts; (2) intermediate releases (builds) of each formal release will be provided for Government use to permit the Government to evaluate progress based upon actual hands-on use of the System.

Three quarters (9 months) are scheduled for the generation of the Type B5 Development Specifications for each formal System release. Each schedule includes not only the actual generation of the Type B5 Development Specifications (event 1) but also an early review of the technical direction being pursued by the Development Phase Contractor (event 2 - Functional Design Review), the final specification of the capabilities to be provided by each build (event 3), and the detailed reviews which culminate in the Government's formal concurrence on a baseline Type B5 Development Specification (event 4).

Based upon Section 4, Quality Assurance Provisions, of the baseline Type B5 Development Specification for each release, an integration and acceptance test plan is generated by the Development Phase Contractor (event 5) and submitted to the Government for concurrence (event 6).

The development of the Type C5 Product Specifications (event 7) for each System release begins when the corresponding Type B5 Development Specification is baselined (event 6) and continues throughout the Development Phase. The Type C5 Product Specification for each release is baselined (event 10) when acceptance testing (Formal Qualification Testing) for the release begins. Two types of formal reviews are scheduled while the Type C5 Product Specification for each release is being developed: a Preliminary Design Review (PDR) and three Critical Design Reviews (CDRs).



The PDR for each release (event 8) is scheduled to occur within 3 months after preparations begin on the corresponding Type C5 Product Specification. These two PDRs provide a positive demonstration that the detailed design of each release is proceeding efficiently. Each CDR provides a formal opportunity for the Government to determine whether the functional requirements of the Type B5 Development Specification are being met by the detailed System design, if there is an integrated design with closure, and if the cost and risk factors are reasonable. Acceptance testing specifications and procedures representing the subset of System capabilities provided by each build will also be reviewed at each CDR.

Three CDRs are planned for each System release (event 9). Each CDR is planned to cover all the software in either one or two complete builds. For the First Level Release, the first CDR will cover the first two builds (these two builds are reviewed together because they form the nucleus of the System); the second CDR will cover build 3; and the third CDR will cover builds 4 and 5. For the Second Level Release, each of the three scheduled CDRs will cover one of the remaining three builds (6, 7, and 8).

Implementation and testing of each release will be done incrementally (event 11). Four builds are planned for the First Level Release. The fifth build is a special build which will provide all First Level Release capabilities on CDC 6000/CYBER computers. Three builds (builds 6, 7, 8) are planned for the Second Level Release. As each build is completed (i.e., coding, documentation, integration testing), the corresponding version of the System will be made available to the Government for experimental use so that the Government can both gain early familiarity with the characteristics and potential of the System and evaluate the progress based on actual use of the System.

At the conclusion of implementation and integration testing of the last build in each release (build 4 for the First Level Release and build 8 for the Second Level Release), acceptance testing is scheduled to commence (event 12). Immediately after acceptance testing is completed for each release, a Physical Configuration Audit

(PCA) is conducted (event 13). During the PCA, all documentation produced for the System release being audited is reviewed by the Government to ensure its completeness and accuracy. Following Government acceptance of the Acceptance Testing Report (event 14) and all the documentation pertaining to the System release (event 15), the release will be installed on all Government-specified host computers for validation by the helicopter analysis community.

Table 11 indicates which build contains the capabilities needed to analyze the five technical characteristics (performance, stability and control, loads and vibrations, acoustics, aeroelastic stability) for each of the three life cycle phases (preliminary design, detailed design, research) identified in the Baseline Type A System Specification. The loads and vibrations analysis capability, however, is presented in terms of four categories (rotor loads, airframe loads, engine/drive system loads, and control-system/pilot loads) to show how the total loads and vibrations analysis capability evolves.

Figure 41 represents a model schedule for the development of a CPCI, starting with the generation of the necessary Type C5 Product Specifications. The actual amount of time required to develop a CPCI will vary depending upon the size and complexity of the CPCI. The relative time required for each CPCI development activity is expected to correlate closely with the relative times indicated in the model schedule. A detailed schedule for each specific CPCI will be produced by the contractor or subcontractor responsible for the development of the CPCI.

Tables 12 through 19 present the plan for implementing the software elements of the Operational Complex identified in the Predesign Phase. Each table corresponds to one of the eight builds of the Operational Complex. The software elements in each build table are presented in the hierarchical framework of the Operational Complex so that functional relationships can be understood. In each build table, software elements which are directly related to helicopter analysis (i. e., the Technology Component) are listed ahead of those which are related to

**Table 11. Build Sequence for Achievement of Aircraft Technical Characteristic and Life-Cycle Phase Analysis Capabilities**

<div>LIFE-CYCLE PHASE</div> <div>TECHNICAL CHARACTERISTICS</div>	PRELIMINARY DESIGN	DETAILED DESIGN	RESEARCH
PERFORMANCE	2	4	7
STABILITY AND CONTROL	3	3	7
LOADS AND VIBRATIONS			
ROTOR LOADS	3	3	7
AIRFRAME LOADS	6	7	7
ENGINE/DRIVE SYSTEM LOADS	6	7	7
CONTROL-SYSTEM/PILOT LOADS	6	7	8
ACOUSTICS	4	6	6
AEROELASTIC STABILITY	4	4	7



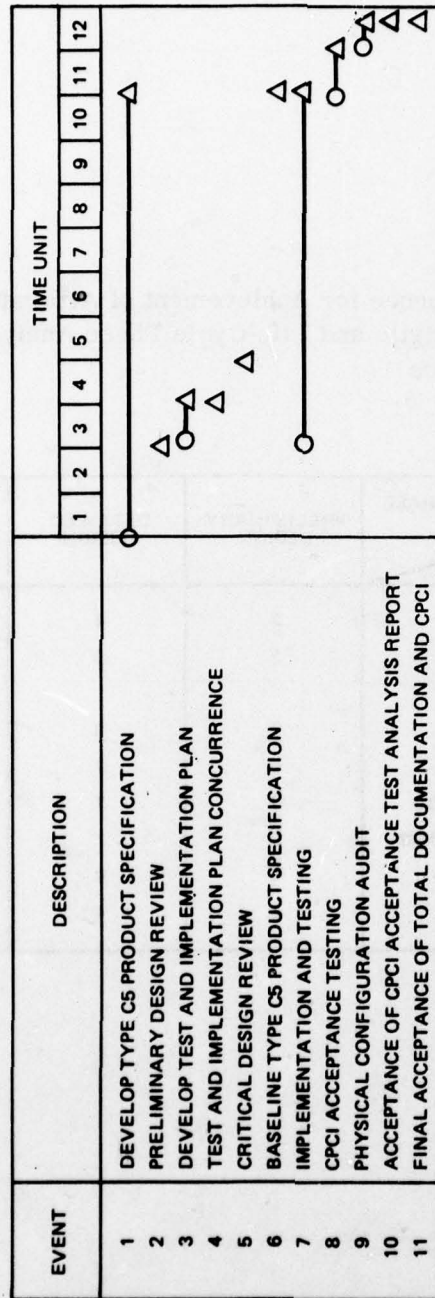


Figure 41. Milestone Schedule for Development of a Typical CPCI of the Operational Complex

Table 12. Build 1 of the Operational Complex (1 of 2)

**FUNCTIONS:**

1. Process a rudimentary set of user input (Case Specification Section)
2. Construct and use a limited Sequence Control Table and a limited Run Data Base
3. Execute dummy Technology Component software elements
4. Identify the software elements to be executed
5. Print user output in a single format
6. Provide computer-independent file management, program management, and storage management support services
7. Provide computer-independent cost assessment and diagnostic support services

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>EXECUTIVE COMPONENT</b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage I	1000	7.5	20K	DPC
2. User Output Subpackage I	1000	7.5	20K	DPC
<b>RUN-TIME MANAGEMENT SUBSYSTEM</b>				
1. Sequence Control Subpackage I	100	0.5	2K	DPC
2. Run-Time Control Package	1000	5.5	20K	DPC
<b>DATA BASE MANAGEMENT SUBSYSTEM</b>				
1. Data Storage Subpackage I	2000	15.0	40K	DPC
2. Data Retrieval Subpackage I	2000	15.0	40K	DPC

- <sup>1</sup> DPC - DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM  
MEMBER SUBCONTRACTOR  
CS - CPCI SUBCONTRACTOR  
GF - GOVERNMENT FURNISHED

Table 12. Build 1 of the Operational Complex (2 of 2)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>OPERATING SYSTEM SERVICE SUBSYSTEM</b>				
1. Host 1 File Management Subpack- age	1000	7.5	20K	DPC
2. Host 1 Program Management Subpackage	1000	7.5	20K	DPC
3. Host 1 Storage Management Sub- package	1000	7.5	20K	DPC
4. Host 1 Cost Assessment and Diagnostic Services Subpackage	1000	5.5	20K	DPC
<b>TOTAL</b>	<b>11100</b>	<b>79.0</b>		



**Table 13. Build 2 of the Operational Complex (1 of 4)**

**ADDED FUNCTIONS:**

1. Perform Preliminary Design Performance Analysis (steady-state and transient)
2. Provide all the general mathematical capabilities needed to perform a steady-state and a transient Preliminary Design Performance Analysis
3. Process an expanded set of user input (Configuration Specification Section and Conditions Specification Section)
4. Allow the specification of all possible System Commands except for those required for restarting an analysis run
5. Create the Run Data Base from the Master Data Base and from user input
6. Provide multiple output report formats
7. Provide System status information whenever a potential internal System software error is detected

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b><u>TECHNOLOGY COMPONENT</u></b>				
<b>SIMULATION MODEL INITIALIZATION SUBSYSTEM</b>				
1. Combine Aircraft Components Package	500	3.0	18K	DPC
2. Combine Environment Compo- nents Package	500	3.0	18K	DPC
3. Combine Aircraft and Environ- ment Components Package	500	3.0	18K	DPC
4. Coordinate Systems and Trans- formations Package	250	1.5	10K	DPC

<sup>1</sup>DPC - DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM  
MEMBER SUBCONTRACTOR

CS - CPCI SUBCONTRACTOR

GF - GOVERNMENT FURNISHED

Table 13. Build 2 of the Operational Complex (2 of 4)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>SIMULATION MODEL SUBSYSTEM</b>				
1. Rigid Blade Rotor Subpackage	400	2.5	12K	DPC
2. Steady Aerodynamic Field for a Rotor Subpackage	150	1.0	5K	DPC
3. Standard Helicopter Rigid Con- trol System Subpackage	100	0.5	3K	DPC
4. Rigid Drive System/Constant Speed Analysis Subpackage	150	1.0	5K	DPC
5. Engine Performance Table Sub- package	200	1.0	6K	DPC
6. Rigid Two-Dimensional Fuse- lage Subpackage	200	1.0	6K	DPC
7. Rigid Three-Dimensional Fuse- lage Subpackage	200	1.0	6K	DPC
8. Aerodynamic Field for an Aero- dynamic Surface Subpackage	150	1.0	5K	DPC
9. Rigid Aerodynamic Surface Subpackage	100	0.5	3K	DPC
10. Rigid Stores Subpackage	200	1.0	6K	DPC
11. Steady Aerodynamic Coefficients Using Simple Equations Sub- package	100	1.0	3K	DPC
12. Steady Aerodynamic Coefficients Using Bivariant Table Sub- package	50	0.5	2K	DPC
13. Steady Aerodynamic Coefficients Using Trivariant Table Sub- package	100	0.5	3K	DPC
14. Steady Aerodynamic Coefficients Using Quadrivariant Table Sub- package	150	1.0	5K	DPC
15. Momentum Theory Flow Field Subpackage	150	1.0	6K	CS
16. Momentum Theory Flow Field With Time Delay Subpackage	300	1.5	11K	CS

Table 13. Build 2 of the Operational Complex (3 of 4)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>TRIM SOLUTION SUBSYSTEM</b>				
1. Simultaneous Iterate-to-Trim Subpackage	1000	7.5	30K	DPC
<b>MANEUVER SUBSYSTEM</b>				
1. Prescribed Control Motions Package	300	1.5	10K	DPC
2. Prescribed Aircraft Response Package	400	3.0	12K	DPC
<b>GENERAL MATHEMATICAL OPERATIONS SUBSYSTEM</b>				
1. Linear Algebraic Equation Solution Package	800	4.5	30K	DPC
2. Matrix Decomposition Package	600	3.5	25K	DPC
3. Differential Equation Solution Package	1000	5.5	40K	DPC
4. Quadrature Package	200	1.0	8K	DPC
5. Matrix Multiplication and Addition Package	500	3.0	20K	DPC
6. Nonlinear Algebraic Equation Solution Package	600	3.5	25K	DPC
7. General Coordinate Transformation Package	100	0.5	4K	DPC
8. Numerical Differentiation Package	600	3.5	25K	DPC
9. Interpolation/Extrapolation Package	400	2.5	15K	DPC
10. Harmonic Analysis Package	400	2.5	15K	DPC
<b><u>EXECUTIVE COMPONENT</u></b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage II	2000	15.0	40K	DPC
2. User Output Subpackage II	1000	7.5	20K	DPC



**Table 13. Build 2 of the Operational Complex (4 of 4)**

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>RUN-TIME MANAGEMENT SUBSYSTEM</b>				
1. Sequence Control Subpackage II	100	0.5	2K	DPC
2. Internal System Error Analysis Package	1000	7.5	20K	DPC
<b>DATA BASE MANAGEMENT SUBSYSTEM</b>				
1. Data Storage Subpackage II	1000	5.5	20K	DPC
2. Data Retrieval Subpackage II	1000	5.5	20K	DPC
<b>TOTAL</b>	<b>17450</b>	<b>110.0</b>		

**Table 14. Build 3 of the Operational Complex ( 1 of 3)**

**ADDED FUNCTIONS:**

1. Perform Preliminary Design Stability and Control Analysis
2. Perform Preliminary Design Rotor Loads Analysis
3. Perform Detailed Design Rotor Loads Analysis
4. Perform Detailed Design Stability and Control Analysis
5. Allow the specification of unsteady airloads
6. Permit the definition of moveable control surfaces
7. Solve the eigenproblem (eigenvalues and eigenvectors)
8. Process the Options Specification Section of user input
9. Restart an analysis run
10. Allow the specification of all possible System Commands
11. Generate a plot in a single format
12. Generate a cost assessment report at run conclusion
13. Maintain and use a partially core-resident Run Data Base

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>TECHNOLOGY COMPONENT</b>				
<b>SIMULATION MODEL INITIALIZATION SUBSYSTEM</b>				
1. Rotor Finite Element Initialization Package	300	1.5	9K	CS
2. Rotor Modes Package	500	3.0	15K	CS
<b>SIMULATION MODEL SUBSYSTEM</b>				
1. Rotor Map Subpackage	200	1.0	6K	DPC
2. Semi-Empirical Rotor Equations Subpackage	250	1.5	9K	DPC
3. Semi-Empirical Directed Fan Subpackage	250	2.0	9K	CS

<sup>1</sup>DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM  
MEMBER SUBCONTRACTOR

CS = CPCI SUBCONTRACTOR

GF = GOVERNMENT FURNISHED

Table 14. Build 3 of the Operational Complex (2 of 3)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
SIMULATION MODEL SUBSYSTEM (Continued)				
4. Dynamic Blade Rotor With Teeter- ing Gimbaled Hub Subpackage	1000	7.5	30K	GF
5. Dynamic Blade Rotor With Artic- ulated Hub Subpackage	1000	7.5	30K	GF
6. Dynamic Blade Rotor With Hingeless Hub Subpackage	1000	7.5	30K	GF
7. Dynamic Blade Rotor With Bear- ingless Hub Subpackage	1000	7.5	30K	GF
8. Rotor Loads and Vibrations Subpackage	800	6.0	24K	DPC
9. Unsteady Aerodynamic Field For a Rotor Subpackage	100	0.5	3K	CS
10. Viscous or Hydraulic Lag Damper Subpackage	150	1.0	5K	DPC
11. Elastomeric Lag Damper Subpackage	150	1.0	5K	DPC
12. Rotor Flapping Stops Subpackage	150	1.0	5K	DPC
13. Rotor Lag Stops Subpackage	150	1.0	5K	DPC
14. Auxiliary Controls Subpackage	200	1.0	6K	DPC
15. Rigid Pylon Subpackage	200	1.0	6K	DPC
16. Simple Landing Gear Subpackage	500	3.0	15K	DPC
17. Internal Cargo Subpackage	100	0.5	3K	DPC
18. Unsteady Airloads by Theodorsen/Loewy Theory Sub- package	300	2.0	9K	CS
19. Unsteady Airloads by $\alpha$ , A, B Table Subpackage	400	3.0	12K	CS
20. Flap Aerodynamic Coefficients Subpackage	250	1.5	9K	DPC
21. Spoiler Aerodynamic Coefficients Subpackage	150	1.0	6K	DPC
22. Structural Coupling Package	2000	15.0	60K	DPC



Table 14. Build 3 of the Operational Complex (3 of 3)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>STABILITY AND CONTROL SUBSYSTEM</b>				
1. Linearize Equations for Stability and Control Package	500	3.0	25K	DPC
2. Stability Eigenvalues and Eigenvectors Package	250	1.5	13K	DPC
3. Transfer Functions and Frequency Response Package	250	1.5	13K	DPC
<b>GENERAL MATHEMATICAL OPERATIONS SUBSYSTEM</b>				
1. Eigenvalue/Eigenvector Package	1200	7.0	60K	DPC
<b>EXECUTIVE COMPONENT</b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage III	1000	7.5	20K	DPC
2. User Output Subpackage III	1000	7.5	20K	DPC
<b>RUN-TIME MANAGEMENT SUBSYSTEM</b>				
1. Sequence Control Subpackage III	100	0.5	2K	DPC
2. Checkpoint Package	1000	5.5	20K	DPC
<b>DATA BASE MANAGEMENT SUBSYSTEM</b>				
1. Data Storage Subpackage III	2000	11.5	40K	DPC
2. Data Retrieval Subpackage III	2000	11.5	40K	DPC
<b>TOTAL</b>	<b>20400</b>	<b>134.5</b>		

Table 15. Build 4\* of the Operational Complex (1 of 3)

ADDED FUNCTIONS:

1.	Perform Preliminary Design Acoustics Analysis
2.	Perform Preliminary Design Aeroelastic Stability Analysis
3.	Perform Detailed Design Performance Analysis
4.	Perform Detailed Design Aeroelastic Stability Analysis
5.	Perform an analysis of moving deck (on a ship or in ground contact) operations
6.	Provide an alternative trim procedure
7.	Provide a generalized control system model
8.	Provide a whirl/test stand model
9.	Perform wake analysis
10.	Provide an interface for external models
11.	Complete the set of general mathematical operations
12.	Process the Failure/Damage Specification Section of user input
13.	Include a description of failure/damage effects in the Run Data Base (from user input and the Master Data Base)
14.	Generate plots in multiple formats

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>TECHNOLOGY COMPONENT</b>				
<b>SIMULATION MODEL INITIALIZATION SUBSYSTEM</b>				
1. Wake Initialization Package	100	0.5	4K	GF
<b>SIMULATION MODEL SUBSYSTEM</b>				
1. Generalized Coupling Rigid Control System Subpackage	200	1.0	6K	DPC
2. Static Elastic Driveshaft Subpackage	100	0.5	3K	DPC

\* BUILD 4 IS EQUIVALENT TO THE FIRST LEVEL RELEASE CAPABILITY OPERATING ON A HOST 1 COMPUTER.

<sup>1</sup> DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM  
MEMBER SUBCONTRACTOR

CS = CPCI SUBCONTRACTOR

GF = GOVERNMENT FURNISHED

Table 15. Build 4\* of the Operational Complex (2 of 3)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>SIMULATION MODEL SUBSYSTEM</b> (Continued)				
3. Prescribed Motion Test Stand Subpackage	200	1.0	6K	DPC
4. Prescribed Wake Subpackage	1500	8.5	55K	GF
5. Prescribed Motion Ground/ Deck Surface Subpackage	150	1.0	6K	DPC
6. Two-Dimensional Ground/Deck Surface Subpackage	100	0.5	4K	DPC
7. Three-Dimensional Ground/ Deck Surface Subpackage	200	1.0	8K	DPC
<b>TRIM SOLUTION SUBSYSTEM</b>				
1. Fly-to-Trim Package	300	2.5	9K	DPC
<b>AEROELASTIC STABILITY SUBSYSTEM</b>				
1. Linear Aeroelastic Stability Analysis Package	600	4.5	30K	CS
2. Floquet Analysis Package	800	6.0	35K	CS
3. Aeroelastic Stability Post- processing Package	200	1.0	6K	CS
<b>ACOUSTICS SUBSYSTEM</b>				
1. Sound Propagation Package	500	4.0	15K	CS
<b>GENERAL MATHEMATICAL OPERATIONS SUBSYSTEM</b>				
1. Matrix Inversion Package	800	4.5	40K	DPC
2. Moving Block Fast Fourier Transform Package	400	2.5	20K	DPC
3. Prony's Method Package	600	3.5	30K	DPC
4. Statistical Functions Package	400	2.5	20K	DPC



Table 15. Build 4\* of the Operational Complex (3 of 3)

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>EXTERNAL MODELS INTERFACE SUBSYSTEM</b>				
1. External Model I Package	100	0.5	3K	DPC
<b><u>EXECUTIVE COMPONENT</u></b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage IV	1000	5.5	20K	DPC
2. User Output Subpackage IV	1000	5.5	20K	DPC
<b>TOTAL</b>	<b>9250</b>	<b>56.5</b>		

Table 16. Build 5\* of the Operational Complex

ADDED FUNCTION:

Provide the First Level Release capability on the Host 2 Computer family

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b><u>EXECUTIVE COMPONENT</u></b>				
<b>OPERATING SYSTEM SERVICE SUBSYSTEM</b>				
1. Host 2 File Management Subpackage	500	4.0	20K	DPC
2. Host 2 Program Management Subpackage	500	4.0	20K	DPC
3. Host 2 Storage Management Subpackage	500	4.0	20K	DPC
4. Host 2 Cost Assessment and Diagnostic Services Subpackage	500	3.0	20K	DPC
<b>TOTAL</b>	<b>2000</b>	<b>15.0</b>		

\* BUILD 5 IS EQUIVALENT TO THE FIRST LEVEL RELEASE CAPABILITY (BUILDS 1 TO 4) OPERATING ON A HOST 2 COMPUTER

<sup>1</sup> DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM  
MEMBER SUBCONTRACTOR

CS = CPCI SUBCONTRACTOR

GF = GOVERNMENT FURNISHED

Table 17. Build 6 of the Operational Complex (1 of 4)

ADDED FUNCTIONS:

1. Perform Preliminary Design Airframe Loads Analysis
2. Perform Preliminary Design Engine/Drive System Loads Analysis
3. Perform Preliminary Design Control System/Pilot Loads Analysis
4. Perform Detailed Design Acoustics Analysis
5. Perform Research Performance Analysis
6. Perform Research Acoustics Analysis
7. Expand the Detailed Design Rotor Loads Analysis Capability
8. Provide an initial circulation control rotor capability
9. Provide for auxiliary propulsion
10. Expand the wake analysis capability
11. Provide a rotor-to-rotor and rotor-to-aerodynamic-surface analysis capability
12. Allow the user to define output report formats

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXECUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b><u>TECHNOLOGY COMPONENT</u></b>				
<b>SIMULATION MODEL SUBSYSTEM</b>				
1. Elastic Substructured Rotor Analysis Subpackage	1200	9.0	60K	GF
2. Semi-Empirical Circulation Control Rotor Subpackage	250	1.5	8K	GF

<sup>1</sup>DPC - DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM MEMBER SUBCONTRACTOR

CS - CPCI SUBCONTRACTOR

GF - GOVERNMENT FURNISHED



Table 17. Build 6 of the Operational Complex (2 of 4)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
SIMULATION MODEL SUBSYSTEM (Continued)				
3. Semi-Empirical Reaction Drive Rotor Subpackage	250	1.5	8K	GF
4. Rotor Out-of-Plane Pendulum Subpackage	200	1.0	6K	DPC
5. Rotor In-Plane Pendulum Subpackage	200	1.0	6K	DPC
6. Rotor Control Load Reduction Devices Subpackage	200	1.0	6K	DPC
7. Engine Governor and Fuel Control Subpackage	500	3.0	15K	GF
8. Auxiliary Propulsion Subpackage	100	0.5	3K	DPC
9. Rigid Gearbox Subpackage	200	1.0	6K	DPC
10. Static Elastic Torsion Gearbox Subpackage	300	1.5	9K	DPC
11. Static Elastic Drivebelt Subpackage	100	0.5	3K	GF
12. Clutch Analysis Subpackage	200	1.0	6K	DPC
13. Drive System Loads and Vibra- tions Subpackage	1000	5.5	30K	DPC
14. Dynamic Test Stand Subpackage	400	2.5	12K	DPC
15. Static Elastic Pylon Subpackage	300	1.5	9K	DPC
16. Static Elastic Aerodynamic Surface Subpackage	200	1.0	6K	DPC
17. Rigid Suspended Cargo Subpackage	200	1.0	6K	DPC
18. Vibration Control Devices Subpackage	300	1.5	9K	DPC
19. Fuel Subpackage	400	2.5	12K	DPC
20. Free Wake Subpackage	3000	17.0	110K	GF
21. Table Look-Up Flow Field Subpackage	500	3.0	15K	DPC

Table 17. Build 6 of the Operational Complex (3 of 4)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>SIMULATION MODEL SUBSYSTEM</b> (Continued)				
22. Rotor-to-Rotor Interference Subpackage	500	4.0	20K	GF
23. Rotor-to-Aerodynamic Surface Interference Subpackage	500	4.0	20K	GF
24. Nonrotating Aerodynamic Surface Potential Flow Subpackage	3000	22.5	90K	GF
25. Semi-Empirical Circulation Control Aerodynamics Subpackage	150	1.0	5K	GF
26. Analytical Wind Tunnel Boundary Conditions Subpackage	1500	17.0	45K	GF
27. Empirical Wind Tunnel Boundary Conditions Subpackage	250	2.0	8K	GF
<b>TRIM SOLUTION SUBSYSTEM</b>				
1. Decoupled Iterate-to-Trim Package	1000	7.5	30K	DPC
<b>MANEUVER SUBSYSTEM</b>				
1. Gust Response Package	400	2.5	12K	DPC
2. Initiate Failure/Damage Effects Package	150	1.0	5K	DPC
<b>ACOUSTICS SUBSYSTEM</b>				
1. Rotor Rotational Sound Subpackage	1000	7.5	30K	GF
2. Rotor Broadband Sound Subpackage	200	1.5	6K	GF
3. Reciprocating Engine Sound Subpackage	200	1.5	6K	GF
4. Turbine Engine Sound Subpackage	250	2.0	8K	GF

Table 17. Build 6 of the Operational Complex (4 of 4)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>ACOUSTICS SUBSYSTEM</b> (Continued)				
5. Gearbox Sound Package	250	2.0	8K	GF
6. Accessories Sound Package	500	4.0	15K	GF
<b><u>EXECUTIVE COMPONENT</u></b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage V	1000	5.5	20K	DPC
2. User Output Subpackage V	1000	5.5	20K	DPC
<b>TOTAL</b>	<b>21850</b>	<b>149.0</b>		



**Table 18. Build 7 of the Operational Complex (1 of 4)**

**ADDED FUNCTIONS:**

1. Perform Detailed Design Airframe Loads Analysis
2. Provide Detailed Design Engine/Drive System Loads Analysis
3. Provide Detailed Design Control System/Pilot Loads Analysis
4. Perform Research Performance Analysis
5. Perform Research Stability and Control Analysis
6. Perform Research Rotor Loads Analysis
7. Perform Research Airframe Loads Analysis
8. Perform Research Engine/Drive System Loads Analysis
9. Perform Research Aeroelastic Stability Analysis
10. Provide dynamic teetering/gimbale rotor analysis
11. Provide more detailed control system/pilot, engine/drive system, and airmass models
12. Provide more complex ground/deck surface models
13. Allow the user to specify alternative analysis techniques
14. Process the Accuracy Assessment Section of user input
15. Provide the capability to assess the accuracy of an analysis
16. Generate cost prediction information for an analysis run
17. Provide a direct interface to the user on an interactive terminal for input data preparation and output data inspection
18. Provide an interactive tutorial capability to assist the user in preparing valid input data

Table 18. Build 7 of the Operational Complex (2 of 4)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b><u>TECHNOLOGY COMPONENT</u></b>				
<b>SIMULATION MODEL SUBSYSTEM</b>				
1. Rotor Servo Flaps Subpackage	200	1.0	6K	GF
2. Static Elastic Control System Subpackage	300	2.5	9K	DPC
3. Dynamic Control System Subpackage	300	2.5	9K	DPC
4. Control System Loads and Vibrations Subpackage	600	4.5	20K	DPC
5. Force Feel System Subpackage	800	6.0	25K	DPC
6. Simple Circulation Control Rotor Control System Subpackage	100	1.0	3K	GF
7. Engine Manufacturers Simulation Subpackage	500	4.0	15K	GF
8. Detailed Engine Analysis Subpackage	500	4.0	15K	GF
9. Reaction Drive Subpackage	200	1.5	6K	GF
10. Circulation Control Drive Subpackage	300	2.5	9K	GF
11. Dynamic Torsion and Bending Driveshaft Subpackage	300	2.5	9K	DPC
12. Dynamic Torsion Gearbox Subpackage	500	4.0	15K	DPC
13. Dynamic Torsion Drive Belt Subpackage	300	2.5	9K	DPC
14. Dynamic Fuselage/Airframe Subpackage	1000	7.5	30K	DPC

<sup>1</sup> DPC - DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM MEMBER SUBCONTRACTOR

CS - CPCI SUBCONTRACTOR

GF - GOVERNMENT FURNISHED

Table 18. Build 7 of the Operational Complex (3 of 4)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>SIMULATION MODEL SUBSYSTEM</b> (Continued)				
15. Dynamic Pylon Subpackage	600	4.5	18K	DPC
16. Dynamic Aerodynamic Surface Subpackage	400	3.0	12K	DPC
17. Detailed Landing Gear Subpackage	700	5.0	21K	DPC
18. Dynamic Suspended Cargo Subpackage	600	4.5	18K	GF
19. Cable Subpackage	400	3.0	12K	GF
20. Suspended Cargo Stabilization Devices Subpackage	300	2.5	9K	GF
21. Hoist and Load Isolation Subpackage	200	1.5	6K	GF
22. Airframe Loads and Vibrations Subpackage	500	4.0	15K	DPC
23. Dynamic Stores Subpackage	500	4.0	15K	DPC
24. Fuselage to Aerodynamic Surface Interference Subpackage	1000	7.5	36K	GF
25. Aerodynamic Panel Method for Arbitrary Bodies Subpackage	4000	30.0	120K	GF
26. Cable Aerodynamics Subpackage	100	1.0	3K	GF
27. Dynamic Elastic Ground/Deck Surface Subpackage	300	2.5	11K	DPC
28. Elastic/Plastic Ground/Deck Surface Subpackage	400	3.0	15K	DPC
29. Water Surface Subpackage	400	3.0	15K	DPC
<b>ACCURACY ASSESSMENT SUBSYSTEM</b>				
1. Set Up Accuracy Assessment Cases Package	700	8.0	21K	GF
2. Compute Sensitivity Factors Package	500	6.0	15K	GF



Table 18. Build 7 of the Operational Complex (4 of 4)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>ACCURACY ASSESSMENT SUBSYSTEM (Continued)</b>				
3. Generate Expected Values and Ranges Package	500	6.0	15K	GF
4. Compare Computed Values Versus Experimental Data Package	400	4.5	12K	GF
<b><u>EXECUTIVE COMPONENT</u></b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage VI	1000	7.5	20K	DPC
2. User Output Subpackage VI	1000	7.5	20K	DPC
3. Interactive Terminal Package	10000	76.0	200K	DPC
<b>OPERATING SYSTEM SERVICES SUB- SYSTEM</b>				
1. Host 1 Interactive Terminal Management Subpackage	1000	7.5	20K	DPC
<b>TOTAL</b>	<b>31400</b>	<b>248.0</b>		

Table 19. Build 8\* of the Operational Complex (1 of 2)

ADDED FUNCTIONS:

1. Perform Research Control System/Pilot Loads Analysis
2. Provide a reaction drive rotor model
3. Provide automatic flight control system, control feedback, and pilot transfer models
4. Provide a generalized aerodynamic interference analysis model
5. Allow the user to describe desired plot formats

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>TECHNOLOGY COMPONENT</b>				
<b>SIMULATION MODEL SUBSYSTEM</b>				
1. Reaction Drive Rotor Subpackage	500	4.0	15K	GF
2. Automatic Flight Control System Subpackage	500	4.0	15K	GF
3. Control Feedback From Force/Motion Sensors Subpackage	400	3.0	12K	GF
4. Pilot Transfer Function Subpackage	500	4.0	15K	GF
5. Unsteady Airloads With Time Delay Subpackage	200	2.5	8K	GF
6. General Purpose Aerodynamic Interference Subpackage	2000	23.0	75K	GF

\* BUILD 8 IS EQUIVALENT TO THE SECOND LEVEL RELEASE CAPABILITY

<sup>1</sup> DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM MEMBER SUBCONTRACTOR

CS = CPCI SUBCONTRACTOR

GF = GOVERNMENT FURNISHED

Table 19. Build 8\* of the Operational Complex (2 of 2)

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b><u>EXECUTIVE COMPONENT</u></b>				
<b>USER INTERFACE SUBSYSTEM</b>				
1. User Input Subpackage VII	1000	5.5	20K	DPC
2. User Output Subpackage VII	1000	5.5	20K	DPC
<b>OPERATING SYSTEM SERVICE SUB- SYSTEM</b>				
1. Host 2 Interactive Terminal Management Subpackage	500	4.0	20K	DPC
<b>TOTAL</b>	<b>6600</b>	<b>55.5</b>		



the executive control of the System and the user interface with the System (i.e., the Executive Component). Within each component, software elements are ordered by subsystem, where a subsystem represents a collection of related capabilities.

#### 5.4.2 Implementation Plan for the Support Complex

Four builds are planned for the Support Complex. The first three builds (A, B, C) provide all the automated tools needed to support the development, test, configuration control, release, documentation, and maintenance of the System software and data on a Host 1 computer (IBM S/370, IBM S/360). The fourth build (Build D) provides similar tools on a Host 2 computer (CDC 6000, CDC CYBER).

Figure 42 presents the planned schedule for developing the Support Complex. The schedule indicates that the Support Complex will be available for Government acceptance testing 18 months into the Development Phase. However, individual software elements of the Support Complex will be made available to Operational Complex CPCI subcontractors as each software element is completed.

The Support Complex development schedule presents the same 15 milestone events as were presented in Figure 40 for the Operational Complex. Having the same milestone events ensures that the visibility afforded the Government into the Support Complex development efforts equals the visibility provided into the Operational Complex development efforts. Each event identified in the Support Complex development schedule serves the same function as the corresponding event in the Operational Complex development schedule. To augment the visibility afforded by these 15 events, the intermediate releases (builds) of the Support Complex will be provided for Government use, thus permitting the Government to assess progress based on actual System use.

Because the Support Complex provides support tools needed to develop Operational Complex software, the capabilities in and the schedule for each Support Complex build must be time-phased with the Operational Complex builds.

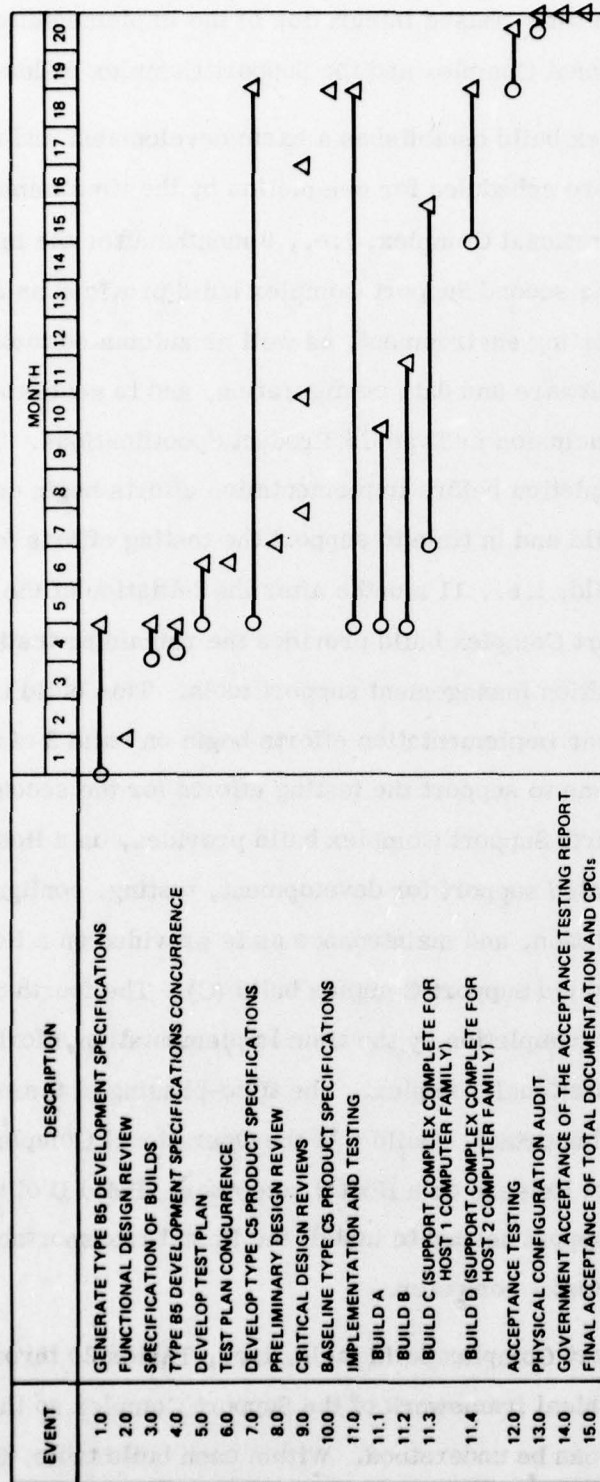


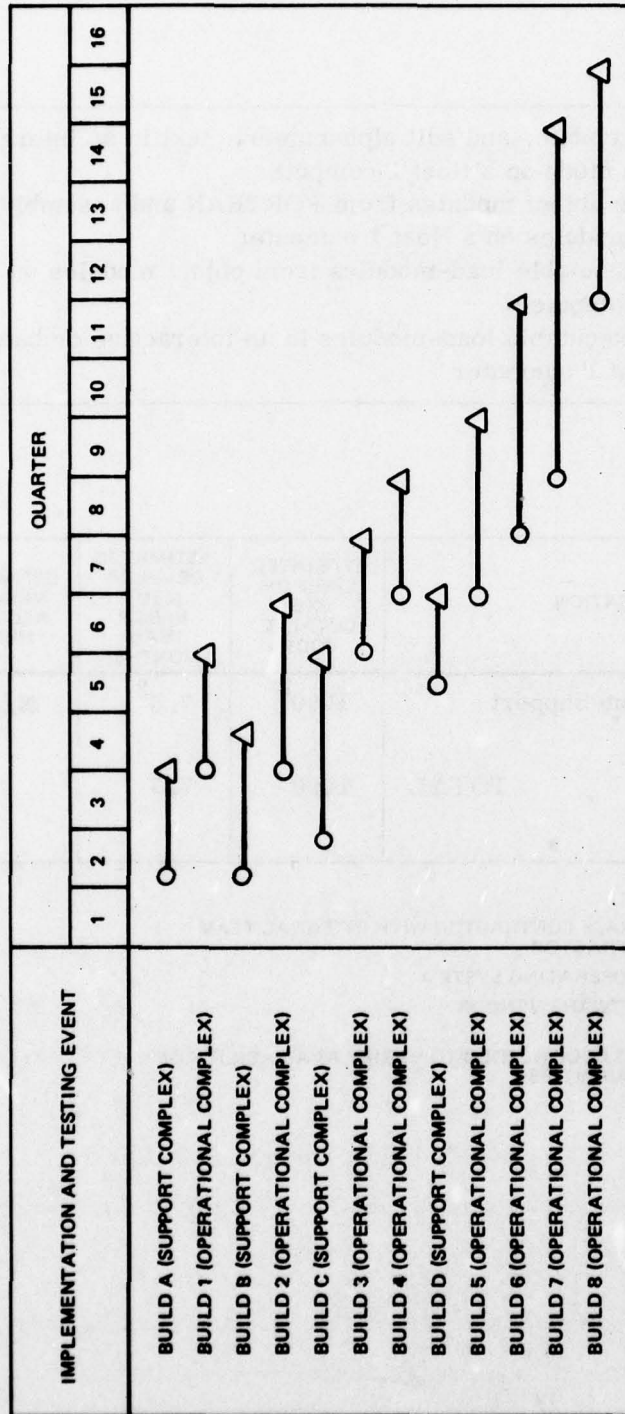
Figure 42. Milestone Schedule for Development of the Support Complex

Figure 43 illustrates the time-phased integration of the implementation and testing schedules of the Operational Complex and the Support Complex builds.

The first Support Complex build establishes a basic development and testing environment and is therefore scheduled for completion by the time implementation efforts begin on the Operational Complex, i.e., 9 months after the initiation of the Development Phase. The second Support Complex build provides an enhanced software development and testing environment, as well as automated tools needed both to control the System software and data configuration, and to generate module design specifications for inclusion in Type C5 Product Specifications. The second build is planned for completion before implementation efforts begin on the second Operational Complex build and in time to support the testing efforts for the first Operational Complex build, i.e., 11 months after the initiation of the Development Phase. The third Support Complex build provides the remaining testing, documentation, and configuration management support tools. This build is planned for completion by the time implementation efforts begin on build 3 of the Operational Complex and in time to support the testing efforts for the second Operational Complex build. The fourth Support Complex build provides, on a Host 2 computer, the same level of automated support for development, testing, configuration control, release, documentation, and maintenance as is provided on a Host 1 computer at the conclusion of the third Support Complex build (C). The fourth Support Complex build is planned for completion by the time implementation efforts begin on the fifth build of the Operational Complex. The time-phasing of these latter two build schedules is very important. Build 5 of the Operational Complex involves installing the First Level Release on a Host 2 computer. Build D of the Support Complex provides the support needed to install the Host 1 transportable System software and data on a Host 2 computer.

The CPCIs in each Support Complex build table, i.e., Tables 20 through 23, are presented in the hierarchical framework of the Support Complex so that CPI functional relationships can be understood. Within each build table, CPCIs are





**Figure 43. Time-Phased Integration of the Operational Complex and Support Complex Implementation Schedules**

Table 20. Build A of the Support Complex

**FUNCTIONS:**

1. Create, update, and edit alphanumeric text in an interactive or batch mode on a Host 1 computer
2. Generate object modules from FORTRAN and assembly language source modules on a Host 1 computer
3. Build executable load-modules from object modules on a Host 1 computer
4. Debug executable load-modules in an interactive or batch mode on a Host 1 computer

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
Host 1 Operating System Support	1000 <sup>2</sup>	7.5 <sup>2</sup>	N/A	HOS
TOTAL	1000	7.5		

<sup>1</sup> DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRAL TEAM MEMBER SUBCONTRACTOR

HOS = HOST COMPUTER OPERATING SYSTEM

CSV = COMMERCIAL SOFTWARE VENDOR

<sup>2</sup> THIS ESTIMATE REFLECTS CODE NEEDED TO VERIFY AVAILABILITY OF OPERATING SYSTEM CAPABILITIES.

Table 21. Build B of the Support Complex (1 of 2)

ADDED FUNCTIONS:

1. Allow the use of structured-programming control statement extensions to ANSI (X3.9-1966) FORTRAN
2. Assess conformance of source modules to programming standards
3. Monitor scope of module, CPCI, and integration tests
4. Control the System software and data configuration on Host 1 computers
5. Generate module design specifications directly from the prologs in source modules

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>DEVELOPMENT SUPPORT SUBSYSTEM</b>				
1. Structured Preprocessor Package	1500 <sup>2</sup>	11.5 <sup>2</sup>	50K	CSV
2. Automated Code Auditor Package	2000	15.0	70K	DPC
<b>TESTING SUPPORT SUBSYSTEM</b>				
1. Decision Path Monitor Package	3000 <sup>2</sup>	22.5 <sup>2</sup>	100K	CSV
<b>CONFIGURATION MANAGEMENT SUPPORT SUBSYSTEM</b>				
1. Host 1 Configuration Control Subpackage	2000	15.0	70K	DPC

<sup>1</sup>DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRATED TEAM  
MEMBER SUBCONTRACTOR

CSV = COMMERCIAL SOFTWARE VENDOR

GF = GOVERNMENT FURNISHED

<sup>2</sup>THIS ESTIMATE NOT INCLUDED IN THE BUILD TOTAL BECAUSE PURCHASE/LEASE IS RECOMMENDED



**Table 21. Build B of the Support Complex (2 of 2)**

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>DOCUMENTATION SUPPORT SUBSYSTEM</b>				
<b>1. Module Specifications Package</b>	500	3.0	20K	DPC
<b>TOTAL</b>	4500	33.0		

**Table 22. Build C of the Support Complex**

**ADDED FUNCTIONS:**

1. Generate data for testing modules and CPCIs
2. Control the content of the Master Data Base and Master Command File
3. Install the System on Government-specified Host 1 computers
4. Identify module and COMMON block cross-reference for inclusion in the dictionary of computer variables

**CONTENTS:**

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
<b>TESTING SUPPORT SUBSYSTEM</b>				
1. Test Data Generation Package	2000	15.0	70K	DPC
<b>CONFIGURATION MANAGEMENT SUPPORT SUBSYSTEM</b>				
1. Data Base Support Package	4000	30.0	96K	DPC
2. Host 1 System Installation Subpackage	1000	7.5	40K	DPC
<b>DOCUMENTATION SUPPORT SUBSYSTEM</b>				
1. Comprehensive Cross Reference Package	1000	5.5	40K	DPC
<b>TOTAL</b>	<b>8000</b>	<b>58.0</b>		

<sup>1</sup> DPC - DEVELOPMENT PHASE CONTRACTOR WITH INTEGRAL TEAM  
MEMBER SUBCONTRACTOR

HOS - HOST COMPUTER OPERATING SYSTEM

CSV - COMMERCIAL SOFTWARE VENDOR

Table 23. Build D of the Support Complex

ADDED FUNCTIONS:

1. Create, update, and edit alphanumeric text in an interactive or batch mode on a Host 2 computer
2. Generate object modules from FORTRAN and assembly language source modules on a Host 2 computer
3. Build executable load modules from object modules on a Host 2 computer
4. Debug executable load-modules in an interactive or batch mode on a Host 2 computer
5. Transfer transportable System software and data from a Host 1 development computer to a Host 2 development computer
6. Install the System on Government-specified Host 2 computers

CONTENTS:

CPCI IDENTIFICATION	ESTIMATED LINES OF EXE- CUTABLE CODE	ESTIMATED DEVELOP- MENT LABOR (MAN- MONTHS)	ESTIMATED MEMORY REQUIRE- MENTS	CANDIDATE SOFTWARE SOURCE <sup>1</sup>
Host 2 Operating System Support	1000 <sup>2</sup>	7.5 <sup>2</sup>	N/A	HOS
CONFIGURATION MANAGEMENT SUPPORT SUBSYSTEM				
1. Host 2 Configuration Control Subpackage	1000	5.5	70K	DPC
2. Host 2 System Installation Sub- package	1000	7.5	80K	DPC
TOTAL	3000	20.5		

<sup>1</sup>DPC = DEVELOPMENT PHASE CONTRACTOR WITH INTEGRAL TEAM  
MEMBER SUBCONTRACTOR

HOS = HOST COMPUTER OPERATING SYSTEM

CSV = COMMERCIAL SOFTWARE VENDOR

<sup>2</sup>THIS ESTIMATE REFLECTS CODE NEEDED TO VERIFY AVAILABILITY OF  
OPERATING SYSTEM CAPABILITIES.



generally ordered by subsystem, where a subsystem represents a collection of related support capabilities. Two Support Complex CPCIs, the Host 1 Operating System Support CPI and the Host 2 Operating System Support CPI, do not provide capabilities unique to a subsystem. These two special CPCIs provide both development and testing support capabilities.

The candidate source for two Support Complex CPCIs in build B is identified as a "Commercial Software Vendor." These two CPCIs are the Structured Pre-processor CPI and the Decision Path Monitor CPI. Transportable software packages which provide the needed CPI capabilities are available from various commercial software vendors at a cost considerably less than the cost of developing the capabilities expressly for the System. The estimated development labor for these two CPCIs has therefore not been included in the build B table.

## **SECTION 6 - RISK CONSIDERATIONS**

Three risk considerations associated with the System's helicopter analysis capability have been identified (Section 6.1). In addition, four risk considerations associated with the effect of executive overhead on the economy of System utilization have also been identified (Section 6.2).

The risk considerations discussed do not represent inherent deficiencies, rather they represent specific System capabilities and characteristics which could have a major impact on the acceptability of the System if they are not factored into any System design.

For each risk consideration discussed, various alternatives that could increase or decrease risk are identified, evaluation criteria are presented and used to assess the alternatives, and the extent to which each risk consideration has been accounted for is discussed.

### **6.1 HELICOPTER ANALYSIS RISK CONSIDERATIONS**

Three risk areas potentially affecting helicopter analysis have been identified: component coupling (Section 6.1.1), aerodynamic flow field analysis (Section 6.1.2), and numerical integration (Section 6.1.3).

#### **6.1.1 Coupling of Components**

The System should allow the engineering user to define the configuration to be analyzed in terms of the components making up the configuration. Further, the System should allow the engineering user to define and analyze each component independent of its final relationship (i.e., coupling) with other components within the configuration. Without these two capabilities, the engineering user would have to redefine each component as a function of the configuration to be analyzed, thus significantly reducing both the usability and the acceptability of the System within the helicopter analysis community.

Within the non-rotary-wing aerospace community, the importance of these two capabilities, i. e., defining a configuration in terms of its components and defining each component independently, has long been recognized. For this reason, structural analysis programs used within the aerospace industry include these two capabilities, along with systematic and accurate methods for coupling the components (e. g., NASTRAN). None of the first generation helicopter analysis programs includes a general and systematic approach for analyzing a coupled configuration of independently defined components. Because of this lack of experience, component coupling can be considered potentially risky for the Second Generation Comprehensive Helicopter Analysis System. However, the feasibility, utility, and validity of component coupling have been proved elsewhere in the aerospace industry.

There are, however, two other risk factors associated with the application of component coupling technology to rotary-wing aircraft: First, is component coupling feasible when a configuration is composed of both rotating and nonrotating components? Second, which method or methods should be employed to couple components?

Section 2. 1. 3 of this report specifically addresses the feasibility of component coupling for helicopter configurations and numerous state-of-the-art methods for effecting component coupling. Component coupling is feasible for helicopter configurations as long as all transformations associated with rotating coordinate systems are performed within the individual components before components are coupled together.

The method for coupling of components to be used in the System represents the last risk factor associated with component coupling. The selected method must satisfy three important criteria:

- The degrees of freedom in the coupled configuration should be a subset of the degrees of freedom in the uncoupled configuration. (This



reduces the number of degrees of freedom to be carried into subsequent analysis steps, thus reducing the cost of the analysis.)

- The method selected must permit a totally independent analysis of each component. (This allows the engineering user to analyze components separately and then to define a configuration made up of the components.)
- The method selected must be compatible with test procedures commonly used throughout the helicopter industry, both with respect to defining System input which is derived from test results and with respect to using test results to validate the analysis results produced by the System.

Coupling of components can be accomplished by two different methods. The first method is commonly referred to as substructure analysis. One of the earliest comprehensive discussions of this method was presented by Przemieniecki.<sup>10</sup> The second method, commonly called the method of component modes, was first proposed by Hurty.<sup>11, 12</sup> This method and others derived from it have since received much attention.<sup>13 through 25</sup>

In both of these methods, a complex structure is considered to be made up of component substructures connected at specified node points, and the behavior of the structure is determined by considering the behavior of the component substructures at the connection node points. The essential difference between the two methods lies in the different manner in which the behavior of the component substructures is represented. In substructure analysis, the component substructures are represented in terms of mass, damping, and stiffness properties at the connecting node points, whereas, in the component modes method, the substructures are represented by their modal data (eigenvalues and eigenvectors). The chief advantage of the component modes method over the substructure analysis method is that reliable results can normally be obtained using significantly fewer

degrees of freedom than are needed in substructure analysis. This satisfies the first criterion that a method of coupling should satisfy.

The substructure analysis method, because of its reliance on mass, damping, and stiffness properties at connecting node points, is not compatible with test procedures commonly used throughout the helicopter industry. Mass, damping, and stiffness properties at connecting node points cannot be obtained from nondestructive tests. Therefore, System input for the substructure analysis method is rarely derivable from tests. However, System data for the component modes method is derivable from nondestructive tests if the proper component modes approach is used.<sup>22</sup>

Several approaches to the component modes method have been proposed by various investigators. The approach suggested by Hurty,<sup>11,12</sup> and variations of it proposed by others,<sup>14 through 17</sup> employ rigid-body modes, constraint modes, and normal modes with fixed constraints. These approaches are incompatible with helicopter test procedures because the required input data is very difficult to derive from test data and because it is quite difficult to compare analysis results with test results. The approaches proposed by MacNeal<sup>19</sup> and Rubin<sup>20</sup> employ free-body modes and residual effects. Both MacNeal's and Rubin's approaches give good accuracy (this is particularly so for Rubin's approach, which is an improvement over MacNeal's approach) and are compatible with helicopter test procedures; both, however, are restrictive and cumbersome in formulation. More recently, a general component modes approach for inclusion in NASTRAN has been developed by Herting and Hoesly<sup>24</sup> under NASA sponsorship. This approach has the accuracy of Rubin's method<sup>20</sup> without the restriction of having to use specific types of modes. In addition, this general approach obtains the results of all the other methods referenced as special cases. This general approach also satisfies all the three criteria mentioned earlier for a systematic method of coupling components. Therefore, the component modes approach developed for inclusion in NASTRAN is the approach best suited for inclusion in the System.

The CSC/BHT System design can accommodate either of the two dynamic coupling methods mentioned above (i.e., substructure analysis or component modes). However, the funds available for the Development Phase may not permit the implementation of both methods. Because of the advantages offered by the NASTRAN component modes approach, CSC recommends the implementation of this method of coupling for the First Level Release of the System. If funding permits, it is recommended that the substructure analysis method be added to the Second Level Release of the System. It is also recommended that the simultaneous use of both these methods<sup>25</sup> be considered for inclusion in the Long Range System.

#### 6.1.2 Aerodynamic Flow Field Analysis

Aerodynamic flow field analysis is an essential capability within both the fixed-wing industry and the rotary-wing industry. However, the effects of aerodynamic flow fields on aircraft surfaces are considerably more difficult to determine for rotary-wing aircraft than for fixed-wing aircraft. For either type of aircraft, aerodynamic effects can significantly affect the performance, loads, vibration, and stability characteristics of the aircraft. However, aerodynamic flow field effects tend to be more pronounced for rotary-wing aircraft than for fixed-wing aircraft because of the proximity and inherent unsteadiness of the rotor wake. Basic differences between rotary-wing aircraft and fixed-wing aircraft, which illustrate the increased difficulty of aerodynamic flow field analysis on rotary-wing aircraft, include the following:

- The fuselage of a helicopter is relatively unstreamlined compared to most fixed-wing aircraft fuselages. Flow separation is thus a proportionately larger contributor to fuselage drag.
- The main rotor's wake has, at all flight conditions, the possibility of having strong interactions with the main rotor itself, as well as with the tail rotor, fin, fuselage, etc. Thus, the main rotor's wake can contribute dominantly unsteady effects in many flight conditions.



- The main rotor's aerodynamic environment is highly unsteady in forward flight. These unsteady effects may be the same order of magnitude as, or larger than, steady effects.
- Stall may occur on some portion of the rotor in many flight conditions. Viscous flow separation effects are therefore inherently more frequent than for fixed-wing lifting surfaces.
- Rotor free-stream velocities are nonuniform radially for all flight conditions and are also nonuniform azimuthally for forward flight conditions. Also, transonic flow exists on significant portions of the rotor for flight conditions which make up a large part of the helicopter mission profile.
- Wake position, flow separation regions, and rotor blade stall are all strongly dependent on the configuration of the rotary-wing aircraft and flight conditions. Thus, some of the linearizations and resultant superposition of linear solutions employed by the fixed-wing industry for flow field analysis are inherently unsatisfactory for rotary-wing aircraft.

Careful attention must be paid, therefore, to the aerodynamic flow field analysis capability selected for inclusion in the System, in terms of its near-term feasibility, its applicability, its reliability, its generality, and the economy of its utilization. If the selected capability does not achieve all of these characteristics, the acceptability of the System will be affected. The functional characteristics of the generalized flow analysis method to be included in the System must include the following:

- The representations of aerodynamic surfaces must be derivable by the System, using basic aircraft data (e.g., fuselage lines data, blade twist, airfoil type, planform data) and user-specified input data. Advanced aerodynamic panel elements must be available (especially

for large areas of the fuselage side and bottom, tailboom bottom, etc., which may have relatively limited interaction with other aerodynamic surfaces). Methods to optimize panel size need to be developed to reduce computation time yet produce acceptable accuracy.

- The potential flow equations and method of solution must be capable of modeling all rotary-wing flight conditions. The effects of unsteady flow, radially and azimuthally nonuniform rotor free stream, rotor(s) wake, compressibility (subsonic, supersonic, and, if possible, transonic), and rotor blade motions must be treated in a generalized, unified manner. New surface singularity representations are needed to avoid numerical instabilities that often occur when modeling wake interference effects such as blade-vortex interactions.
- Viscous effects (and, if necessary, transonic compressibility effects) must be handled in an approximate manner (to reduce computer costs).

The physical laws that apply to the calculation of aerodynamic forces and moments are represented by the well-known Navier-Stokes nonlinear partial differential equations. Closed-form solutions to the Navier-Stokes equations exist only for a very small number of simplified special cases. Numerical solution techniques for solving the Navier-Stokes equations at acceptable computer costs are not currently well developed. However, complete solutions to the time-dependent Navier-Stokes equations are being attempted at the present time for a variety of

problems.<sup>41,42</sup> Nevertheless, Chapman<sup>43</sup> estimates that it will be the late 1980s before computers will be available that have sufficient speed to solve these equations in a "reasonable" amount of time for fixed-wing problems. This estimate assumes that computer speeds in the late 1980s will be three orders of magnitude faster than the current speed of the Illiac IV. Rotary-wing capabilities will probably lag behind fixed-wing capabilities because there is not a simple, straightforward application of fixed-wing solutions to rotary-wing problems (due to the complicating flow characteristics listed earlier). Therefore, rotary-wing solutions will probably require another order of magnitude increase in computer speed, and Navier-Stokes solutions for rotary wings are not expected to be included within the life cycle of the System.

---

<sup>41</sup>Rudy, D. et al., AN INVESTIGATION OF SEVERAL NUMERICAL PROCEDURES FOR TIME-ASYMPTOTIC COMPRESSIBLE NAVIER-STOKES SOLUTIONS, Paper No. 14 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 437-468.

<sup>42</sup>Thames, F. C., Thompson, J. F., and Mastin, C. W., NUMERICAL SOLUTION OF THE NAVIER-STOKES EQUATIONS FOR ARBITRARY TWO-DIMENSIONAL AIRFOILS, Paper No. 15 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 469-530.

<sup>43</sup>Chapman, D. R., INTRODUCTORY REMARKS, in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 4-7.



A somewhat simpler method would be to solve the following unsteady perturbation potential flow equation<sup>44</sup>:

$$M^{-2} \nabla^2 \Phi - \Phi_{TT} = (\gamma - 1) \nabla^2 \Phi (\Phi_T + (\nabla \Phi)^2 / 2) + \nabla \Phi \cdot \nabla (2\Phi_T + (\nabla \Phi)^2 / 2) \quad (17)$$

where  $M$  is Mach number,  $\Phi$  is the velocity potential function,  $\gamma$  is the ratio of specific heats, and the subscript  $T$  indicates a substantial time derivative operator which, for perturbations about a flow  $U$  in the  $x$ -direction, is given by

$$(\ )_T = U^{-1} \frac{\delta}{\delta t} (\ ) + \frac{\delta}{\delta x} (\ ) \quad (18)$$

and boundary conditions for tangential flow on the surface,  $B(x, y, z, t) = 0$ , is given by

$$B_T + \nabla \Phi \cdot \nabla B = 0 \quad (19)$$

Equation (17) neglects viscous effects. Therefore, an empirical or approximate method is needed to represent flow separation. Some solutions to this complete equation have been obtained, but general and reliable methods are not yet available.

<sup>44</sup>Bland, S. R., RECENT ADVANCES AND CONCEPTS IN UNSTEADY AERODYNAMIC THEORY, Paper No. 46 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 1305-1326.

Still another alternative is to use a simplified form of Equation (17), namely

$$M^{-2} \nabla^2 \Phi - \Phi_{TT} = 0 \quad (20)$$

This equation is equivalent to neglecting transonic flow effects. This form of Equation (17) represents unsteady effects and compressibility effects in a generalized, unified manner. Morino has developed much of this capability for fixed-wing aircraft<sup>45</sup> and has done some work for both tilting prop-rotor aircraft<sup>46</sup> and helicopters.<sup>47</sup> The solution represented by Equation (20) is valid for airframe calculations and for rotor-in-hover calculations.

The provision of an aerodynamic potential flow field analysis capability without transonic effects in the Second Level Release is recommended. This capability is feasible, economical to use, and represents the near-term state of the art.

The absence of a complete unsteady perturbation potential flow capability in the Second Level Release will not affect the acceptability of the System, because such a capability is effectively beyond even the near-term state of the art. However, the provision of a partial unsteady perturbation potential flow capability (i.e., neglecting transonic flow effects) as represented in Equation (20) will enhance the

---

<sup>45</sup> Morino, L. and Chan, L.-T., INDICIAL COMPRESSIBLE POTENTIAL AERODYNAMICS AROUND COMPLEX AIRCRAFT CONFIGURATIONS, Paper No. 38 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 1067-1110.

<sup>46</sup> Morino, L. et al., AERODYNAMIC INTERFERENCE EFFECTS ON TILTING PROPRTOR AIRCRAFT, NASA CR-152053.

<sup>47</sup> Soohoo, P. et al., ROTOR WAKE EFFECTS OF HUB/PYLON FLOW SEPARATION, Aerospace Systems, Inc.; TR 76-38, Vol. 1 (to be published as a Government report, under Contract DAAJ02-75C-0041).

acceptability of the System because it represents a useful and advanced state-of-the-art capability.

It is suggested that the Government incorporate into its plans for the Long Range System the development of a capability to solve the complete unsteady perturbation potential flow problem as expressed in Equation (17). This development effort should begin as soon as reliable solution techniques have been developed for applying Equation (17) to fixed-wing aerodynamic analysis problems. In planning for the Long Range System, the following questions need to be given special consideration:

- Can we afford to wait for the fixed-wing solution?
- When available, will the fixed-wing methods be directly applicable to rotary-wing uses?
- Will independent development of rotary-wing methods be required?

#### 6.1.3 Numerical Integration

The System must provide stable and accurate numerical integration methods if acceptable analyses of flight dynamics problems are to be achieved. Three types of numerical integration methods have been analyzed by CSC/BHT: predictor-corrector methods, Runge-Kutta methods, and direct integration methods.

Predictor-corrector methods, because of their speed, accuracy, and stability, are ideal for many applications. However, BHT's experience with predictor-corrector methods has shown that the high frequencies and forcing function discontinuities typical of many helicopter analysis problems cause even the best predictor-correction methods (e.g., Hamming's method) to be very slow and sometimes inaccurate. Therefore, CSC/BHT does not recommend providing a predictor-corrector numerical integration capability in the System.



Runge-Kutta methods use various weighting functions for averaging the calculated accelerations. Because the methods are self-starting, they are inherently insensitive to discontinuities in the forcing functions. The stability and economy of these methods tend to deteriorate in the presence of high frequencies. Thus, it may be rather time consuming to run cases with very high frequencies. This situation can be improved by the use of enhanced Runge-Kutta methods (e.g., the Runge-Kutta-Fehlberg method), which require fewer function evaluations and allow a variable integration interval with no loss in accuracy or stability. However, stability still tends to deteriorate in the presence of very high frequencies.

CSC/BHT plans to provide an enhanced Runge-Kutta method as one of the available numerical integration methods in the First Level Release. However, the enhanced Runge-Kutta method will not be used for flight dynamics problems in which very high frequencies are present. CSC/BHT plans to use a direct integration method for such cases.

CSC/BHT has studied the applicability of direct integration methods to helicopter flight dynamics problems. Direct integration methods are widely used in the numerical integration of dynamic equilibrium equations encountered in finite element analysis. The term "direct" means that the equations involved are integrated directly without any transformation. This is in contrast to the modal (or mode superposition) approach in which the dynamic equilibrium equations are first transformed (prior to integration) by the use of generalized coordinates.

Direct numerical integration methods employed in practice involve either explicit or implicit formulations. Explicit formulations are based on equilibrium conditions at the previous time step, whereas implicit formulations are based on equilibrium conditions at the current time step. The central difference method is an example of an explicit method, and the Newmark-Beta, Wilson-Theta, and Houbolt methods are examples of implicit methods.<sup>26</sup>

The differences between explicit and implicit methods are quite important. Explicit methods typically require minimal storage and are computationally efficient,

but are hampered because their stability criteria limit the step sizes that can be employed. In general, these step-size limitations are related to the highest natural frequency of the system of equations under consideration. The implicit methods, on the other hand, offer unconditional stability at the expense of increased storage and reduced computational efficiency. This unconditional stability results from the introduction of synthetic damping into the system of equations when very high frequencies are present, thus affecting the accuracy of the solution. This can present a potentially serious problem to the engineer, because the integration results are used for stability analysis. The problem can be resolved if the amount of synthetic damping introduced can be accurately determined by the System. CSC/BHT has not yet found an implicit method possessing this characteristic. However, implicit methods are particularly suitable for large systems of equations where the highest natural frequency is generally not known and may, in fact, approach infinity in certain cases.

CSC and BHT are continuing to investigate numerical integration techniques for application to flight dynamics problems. Recently, an unconditionally stable explicit algorithm has been proposed for certain structural dynamics problems.<sup>27</sup> Other recent publications address the relative stability of various numerical integration techniques for vibration problems<sup>28</sup> and for transient rotor dynamics problems.<sup>29</sup>

## 6.2 EXECUTIVE OVERHEAD CONSIDERATIONS

A consideration associated with every large, general-purpose software system is the overhead associated with executive software. The design of an executive must address two types of overhead: processing overhead and memory overhead. There are two primary areas of concern with respect to executive processing overhead for the System: (1) the amount of time required to interpret a System Command and transfer control to the software element specified by the System Command and (2) the amount of time required to store data in and retrieve data from the Run Data Base. Section 6.2.1 addresses the first concern; Section 6.2.2

addresses the second. Section 6.2.3 discusses the effect of executive processing overhead on the execution time of small problems. Section 6.2.4 addresses executive design concepts which will minimize the amount of computer memory required by executive software.

#### 6.2.1 System Command Execution

To make the System acceptable for use by the helicopter analysis community, the amount of executive processing overhead must not contribute significantly to the total computer time required for an analysis. The processing overhead associated with executing a System Command is equivalent to the time expended from the end of execution of one software element to the beginning of execution of the next software element specified in a System Command. The time expended must be examined relative to the execution time of the software element.

The time expended depends on four operations: (1) interpreting the System Command, (2) bringing the software element into memory if it is not already there, (3) bringing the required data elements into memory if they are not already there, and (4) transferring control to the software element. The amount of time associated with operations (1) and (4) is, relative to operations (2) and (3), insignificant. To reduce the overhead associated with operation (2), CSC's design permits more than one software element per load-module. In addition, once a load-module is in memory it is retained until that section of memory is required. Thus, bringing the software element into memory is the exception rather than the rule and processing overhead is minimized. To reduce the overhead associated with operation (3), CSC has designed the System to allow the Run Data Base to be entirely memory-resident for small problems and to be partially memory-resident for intermediate-size problems. This design decision virtually eliminates this type of processing overhead for small problems and reduces it considerably for intermediate-size problems.

Based on CSC's 7-year experience with the Goddard Real Time System, a software system for real-time processing of satellite launch data that uses a



load-module management philosophy similar to the one designed by CSC for the System, CSC estimates that the processing overhead penalty for the four operations described above for any one software element will amount to no more than 9 percent in the worst case and considerably less than that in the majority of cases. The 9-percent worst case overhead penalty applies not to the whole problem but to one software element only. The processing overhead penalty for a complete analysis run will be considerably less because operations (2) and (3) described above will seldom occur.

There are many advantages to using System Commands to control processing. Some of these are the increased flexibility the user has to view intermediate output by printing and plotting anywhere in the command sequence and to build an unlimited number of System Command Sequences; the ability to improve efficiency if more memory is available; and the ability to gather System performance statistics anywhere in the sequence. These advantages outweigh the processing overhead penalty incurred.

#### 6.2.2 Data Base Management Capability

For the System to be able to accommodate new and improved capability throughout its target lifetime, it is essential to separate the location of the data from the internal logic of software elements. It is also essential that this separation be accomplished in such a way as to minimize the computer time used. Without proper care, executive processing overhead in managing the data base could prevent the System from enjoying widespread use. It is for this reason that CSC has not recommended the use of a generalized off-the-shelf data base management system such as System 2000 or TOTAL. These systems are designed for large data bases of a business-oriented nature and, while very powerful, do penalize the user with a relatively high overhead. Rather, CSC proposes to develop, as part of the Executive Component, a simple data base management capability (as opposed to a general-purpose data base management capability) tailored specifically to the needs of helicopter analysis. CSC's design of this capability,

incorporated in the Data Base Management Subsystem, has the advantage that data elements are retained in memory (the memory-resident Run Data Base) if memory is available. This will be the case for small problems.

### 6.2.3 Small Problems

The primary concern with respect to solving small problems, e.g., those associated with the preliminary design aircraft life cycle phase, can be stated as follows: many design features are required to make the System flexible, extendable, and global in application; these features may not be required for running a small problem and thus the cost of running a small problem might be exorbitant. This has been a common complaint about NASTRAN. However, NASTRAN was specifically designed to solve large problems. This is not the case for the Second Generation Comprehensive Helicopter Analysis System. It is a requirement that the System solve both small and large problems and that small problems be solved efficiently.

It is to be expected that, as problem size decreases, the relative execution time of many of the software elements becomes smaller and thus the percentage of executive processing overhead increases. It is also to be expected that a system of general applicability will require a longer execution time for any individual problem than would a computer program specially designed for that problem. The goal is not to eliminate executive processing overhead for small problems but to limit it within acceptable bounds.

The principal characteristic of NASTRAN that causes it to be inefficient for small problems is that data blocks are stored on and retrieved from external files regardless of the size of the problem being solved. Therefore, storing and retrieving data blocks that are input to and output from NASTRAN functional modules involve physical input/output operations independent of problem size. The fixed overhead penalty associated with these input/output operations experienced by NASTRAN users is avoided in the System by keeping data of the Run Data Base in

memory if memory is available. For small problems, the Run Data Base will be memory-resident, thus eliminating the overhead associated with writing and reading the Run Data Base or parts of it to and from external files. Thus, because of the memory residence of the Run Data Base, small problems can be analyzed at low cost.

#### 6.2.4 Memory Overhead

To make the System acceptable for day-to-day use by the helicopter analysis community, the amount of computer memory required by executive software must not contribute significantly to the total computer memory required for an analysis run. To minimize the memory overhead contribution of executive software, the executive design approach used minimizes the amount of executive software which must be resident in the computer while a software element is executing. Two types of executive services are required by a software element during execution: run-time control services and data base management services.

The Run-Time Control Package includes three sets of run-time capabilities: identifying the input data in the Run Data Base to be used by a software element, intercepting unexpected processing errors occurring during software element execution, and detecting normal termination of software element execution. The only run-time control capabilities which need be memory resident while a software element is executing are the capability to intercept unexpected processing errors occurring during software element execution and the capability to detect normal termination of software element execution. These two run-time control capabilities represent a very small software subset of all run-time control capabilities. Therefore, the memory overhead contribution of the Run Time Control Package is expected to be very small in proportion to the memory requirements of an average analysis software element.

The Data Base Management Subsystem includes two sets of data base management capabilities: reading/writing data from/to the Master Data Base and reading/writing data from/to the Run Data Base. Because a software element will not,



in CSC's design, directly access any data from the Master Data Base, the data base management software that manages the Master Data Base need not be memory resident while a software element is executing. In addition, few technology software elements will require that data be written onto non-memory-resident portions of the Run Data Base, e.g., those software elements that will be generating matrices that exceed the size of data memory available to the analysis run. For small problems, only the data base management services required to access data from the Run Data Base need be memory resident while a software element is executing.

The planned partitioning of run-time control and data base management services into services required prior to software element execution, during software element execution, and after software element execution will minimize the executive software required during software element execution. In this way, the CSC design of the System executive will ensure that the memory overhead of the System executive is minimized and will not contribute significantly to the total computer memory required for an analysis run.

### REFERENCES

1. Kreyszig, E., ADVANCED ENGINEERING MATHEMATICS, New York, John Wiley and Sons, 1967.
2. Seckel, E., STABILITY AND CONTROL OF AIRPLANES AND HELICOPTERS, New York, Academic Press, 1968.
3. Bramwell, A. R. S., HELICOPTER DYNAMICS, New York, John Wiley and Sons, 1976.
4. Desai, C. S., and Abel, J. F., INTRODUCTION TO THE FINITE ELEMENT METHOD, New York, Van Nostrand Reinhold Company, 1972.
5. Brebbia, C. A., and Connor, J. J., FUNDAMENTALS OF FINITE ELEMENT TECHNIQUES, New York, John Wiley and Sons, 1974.
6. Gallagher, R. H., FINITE ELEMENT ANALYSIS FUNDAMENTALS, Englewood Cliffs, New Jersey, Prentice-Hall Inc., 1975.
7. Cronkhite, J. D., DEVELOPMENT, DOCUMENTATION AND CORRELATION OF A NASTRAN VIBRATION MODEL OF THE AH-1G HELICOPTER AIRFRAME, NASTRAN: User's Experiences, NASA TM X-3428, October 1976, pp. 273-294 (see also Reference 8).
8. Pamidi, P. R., and Cronkhite, J. D., ADDITION OF RIGID ELEMENTS TO NASTRAN, Sixth NASTRAN Users' Colloquium, NASA Conference Publication 2018, October 1977, pp. 449-468.
9. Krishna Murthy, A. V., and Sridhara Murthy, S., FINITE ELEMENT ANALYSIS OF ROTORS, Mechanism and Machine Theory, Vol. 12, 1977, pp. 311-322.
10. Przemieniecki, J. S., MATRIX STRUCTURAL ANALYSIS OF SUBSTRUCTURES, AIAA Journal, Vol. 1, No. 1, January 1963, pp. 138-147.
11. Hurty, W. C., VIBRATIONS OF STRUCTURAL SYSTEMS BY COMPONENT MODE SYNTHESIS, Proc. ASCE, Journal of the Engineering Mechanics Division, August 1960, pp. 51-69.
12. Hurty, W. C., DYNAMIC ANALYSIS OF STRUCTURAL SYSTEMS USING COMPONENT MODES, AIAA Journal, Vol. 3, No. 4, April 1965, pp. 678-685.

13. Gladwell, G. M. L., BRANCH MODE ANALYSIS OF VIBRATING SYSTEMS, Journal of Sound and Vibration, Vol. 1, 1964, pp. 41-59.
14. Bamford, R. M., A MODAL COMBINATION PROGRAM FOR DYNAMIC ANALYSIS OF STRUCTURES, NASA TM 33-290, Jet Propulsion Laboratory, 1966.
15. Bajan, R. L., and Feng, C. C., FREE VIBRATION ANALYSIS BY THE MODAL SUBSTITUTION METHOD, American Astronautics Society Symposium, Paper No. 68-8-1, July 1968.
16. Benfield, W. A., and Hruda, F. R., VIBRATION ANALYSIS OF STRUCTURES BY COMPONENT SUBSTITUTION, AIAA Journal, Vol. 9, July 1971, pp. 1255-1261.
17. Hintz, R. M., ANALYTICAL METHODS IN COMPONENT MODAL SYNTHESIS, AIAA Journal, Vol. 13, August 1975, pp. 1007-1016.
18. Craig, R. R., Jr., and Bampton, M. C. C., COUPLING OF SUBSTRUCTURES FOR DYNAMIC ANALYSES, AIAA Journal, Vol. 6, No. 7, July 1968, pp. 1313-1319.
19. MacNeal, R. H., A HYBRID METHOD OF COMPONENT MODE SYNTHESIS, Computers and Structures, Vol. 1, December 1971, pp. 581-601.
20. Rubin, S., AN IMPROVED COMPONENT-MODE REPRESENTATION, AIAA Paper No. 74-386, presented at AIAA/ASME/SAE 15th Structures, Structural Dynamics and Materials Conference, Las Vegas, Nevada, April 17-19, 1974.
21. Gieseke, R. K., ANALYSIS OF NONLINEAR STRUCTURES VIA MODE SYNTHESIS, NASTRAN: Users' Experiences, NASA TM X-3278, September 1975, pp. 341-360.
22. Klosterman, A. L., A COMBINED EXPERIMENTAL AND ANALYTICAL PROCEDURE FOR IMPROVING AUTOMOTIVE SYSTEM DYNAMICS, Society of Automotive Engineers, Paper No. 720093, January 1972.
23. McClelland, W. A., and Klosterman, A. L., USING NASTRAN FOR DYNAMIC ANALYSIS OF VEHICLE SYSTEMS, Society of Automotive Engineers, Paper No. 740326, March 1974.



AD-A063 921

COMPUTER SCIENCES CORP SILVER SPRING MD SYSTEM SCIEN--ETC F/G 1/3  
THE PREDESIGN PHASE OF THE SECOND-GENERATION COMPREHENSIVE HELI--ETC(U)  
OCT 78 DAAJ02-77-C-0057

UNCLASSIFIED

USARTL-TR-78-41

NL

5 OF 5  
AD  
A063921



END  
DATE  
FILMED  
4 -79  
DDC

24. Herting, D. N., and Hoesly, R. L., DEVELOPMENT OF AN AUTOMATED MULTISTAGE MODAL SYNTHESIS SYSTEM FOR NASTRAN, Sixth NASTRAN Users' Colloquium, NASA Conference Publication 2018, October 1977, pp. 435-448.
25. Wilson, T. L., A NASTRAN DMAP ALTER FOR THE COUPLING OF MODAL AND PHYSICAL COORDINATE SUBSTRUCTURES, Sixth NASTRAN Users' Colloquium, NASA Conference Publication 2018, October 1977, 119-130.
26. Bathe, K. J., and Wilson, E. L., NUMERICAL METHODS IN FINITE ELEMENT ANALYSIS, Englewood Cliffs, New Jersey, Prentice-Hall, Inc., 1976.
27. Trujillo, D. M., AN UNCONDITIONALLY STABLE EXPLICIT ALGORITHM FOR STRUCTURAL DYNAMICS, International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1579-1592.
28. Wood, W. L., ON THE ZIENKIEWICZ FOUR-TIME-LEVEL SCHEME FOR THE NUMERICAL INTEGRATION OF VIBRATION PROBLEMS, International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1519-1528.
29. Kascak, A. F., STABILITY OF NUMERICAL INTEGRATION TECHNIQUES FOR TRANSIENT ROTOR DYNAMICS, NASA Technical Paper 1092, November 1977.
30. Hamilton, W. R., ON A NEW SPECIES OF IMAGINARY QUANTITIES CONNECTED WITH A THEORY OF QUATERNIONS, Dublin Proc., Vol. 2, No. 13, November 1843, pp. 424-434.
31. Yang, A. T., APPLICATION OF QUATERNION ALGEBRA AND DUAL NUMBERS TO THE ANALYSIS OF SPATIAL MECHANISMS, doctoral dissertation, Columbia University, New York, 1963.
32. Wertz, J. R. (editor), SPACECRAFT ATTITUDE DETERMINATION AND CONTROL, Dordrecht-Holland D. Reidel, in preparation.
33. Friedmann, P. Hammond, C. E., and Woo, T. H., EFFICIENT NUMERICAL TREATMENT OF PERIODIC SYSTEMS WITH APPLICATION TO STABILITY PROBLEMS, International Journal for Numerical Methods in Engineering, Vol. 11, 1977, pp. 1117-1136.

34. Done, G. T. S., and Simpson, A., DYNAMIC INSTABILITY OF CERTAIN CONSERVATIVE AND NON-CONSERVATIVE SYSTEMS, Journal of Mechanical Engineering Science, Vol. 19, No. 6, 1977, pp. 251-263.
35. PROGRAMMING STANDARDS FOR THE SECOND-GENERATION COMPREHENSIVE HELICOPTER ANALYSIS SYSTEM, Applied Technology Laboratory, U.S. Army Research and Technology Laboratories, Fort Eustis, Virginia, to be published.
36. NOMENCLATURE OF THE SECOND-GENERATION COMPREHENSIVE HELICOPTER ANALYSIS SYSTEM, Applied Technology Laboratory, U.S. Army Research and Technology Laboratories, Fort Eustis, Virginia, to be published.
37. RESEARCH AND DEVELOPMENT SOFTWARE ACQUISITION - A GUIDE FOR THE MATERIEL DEVELOPER, AMCP 70-4, Army Materiel Command, Headquarters U.S. Army Materiel Command, September 1974.
38. SPECIFICATION PRACTICES, MIL-STD-490, U.S. Department of Defense, October 1968.
39. CONFIGURATION MANAGEMENT PRACTICES FOR SYSTEMS, EQUIPMENT, MUNITIONS, AND COMPUTER PROGRAMS, MIL-STD-483, U.S. Department of Defense, December 1970.
40. AUTOMATED DATA SYSTEM DOCUMENTATION STANDARDS MANUAL, DoD Manual 4120.17-M, Department of Defense, December 1972.
41. Rudy, D. H. et al., AN INVESTIGATION OF SEVERAL NUMERICAL PROCEDURES FOR TIME-ASYMPTOTIC COMPRESSIBLE NAVIER-STOKES SOLUTIONS, Paper No. 14 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 437-468.
42. Thames, F. C., Thompson, J. F., and Mastin, C. W., NUMERICAL SOLUTION OF THE NAVIER-STOKES EQUATIONS FOR ARBITRARY TWO-DIMENSIONAL AIRFOILS, Paper No. 15 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 469-530.
43. Chapman, D. R., INTRODUCTORY REMARKS, in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 4-7.



44. Bland, S. R., RECENT ADVANCES AND CONCEPTS IN UNSTEADY AERODYNAMIC THEORY, Paper No. 46 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 1305-1326.
45. Morino, L., and Chan, L.-T., INDICIAL COMPRESSIBLE POTENTIAL AERODYNAMICS AROUND COMPLEX AIRCRAFT CONFIGURATIONS, Paper No. 38 in Aerodynamic Analyses Requiring Advanced Computers (conference held at NASA Langley Research Center, Hampton, Virginia, March 4-6, 1975), pp. 1067-1110.
46. Morino, L. et al., AERODYNAMIC INTERFERENCE EFFECTS ON TILTING PROPROPOTOR AIRCRAFT, NASA CR-152053.
47. Soohoo, P. et al., ROTOR WAKE EFFECTS OF HUB/PYLON FLOW SEPARATION, Aerospace Systems, Inc.; TR 76-38, Vol. 1 (to be published as a Government report, under Contract DAAJ02-75C-0041).

## GLOSSARY

This glossary contains terms used in the CSC/BHT System design. The glossary forms the basis for, and will evolve into, the Project glossary that will be part of the System Development Plan (see Section 5.2.1).

<u>Term</u>	<u>Definition</u>
aerodynamic node point	An aerodynamic node point is a node point at which an aerodynamic load is applied. (see node point)
analysis degrees of freedom	Analysis degrees of freedom are all of the independent degrees of freedom which are elements of the $q$ , $\dot{q}$ , or $\ddot{q}$ vectors in the matrix representation of the System equations of motion.
analysis run	An analysis run is a computer job in which the Operational Complex of the System is executed. An analysis run consists of one or more cases.
array	An array is a collection of named data of a single type identified and managed as a single entity for convenience and efficiency. An array contains data-items which are identified by indices appended to the array name; e.g., $B(1,3)$ denotes the element of the first row and third column of the array B.
complex	The System is made up of two complexes: the Operational Complex and the Support Complex. The Operational Complex is that subset of modules of the System

Term	Definition
complex (cont'd)	<p>used by the analyst to obtain predictions in the following five areas of helicopter analysis: performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability. The Support Complex is that subset of modules of the System needed to support the development, test, configuration management, and documentation of the Operational Complex and to support the overall management of the System. The two complexes are mutually exclusive and exhaustive; i.e., a module of the System is an element of the Operational Complex or of the Support Complex but not an element of both, and every module of the System is an element of one of the complexes.</p>
data element	<p>The term "data element" is used to denote, nonspecifically, a member of the data hierarchy in the Master Data Base or Run Data Base; i.e., a data element may be a data-item, an array, or a set.</p>
data-item	<p>A data-item is the smallest element of named data which is separately identified and managed. Data-items may vary in size and type.</p>
development computer	<p>A development computer is a computer used by the Development Phase Contractor to implement and test the System. There are two development computers: one for the Host 1 computer family, the other for the Host 2 computer family.</p>



Term	Definition
dynamic node point	A dynamic node point is a node point at which motions are defined or are to be calculated or at which dynamic loads are applied. (see node point)
Executive Component file	See subsystem.  A file is a collection of data residing on an external storage device; a file has no implication of a data hierarchy associated with it.
Host 1 computer family	The Host 1 computer family consists of the IBM S/370 Model 158 under OS/VS, the IBM S/370 Model 168 under OS/VS, and the IBM S/360 Model 65 under OS/MVT.
Host 2 computer family	The Host 2 computer family consists of the CDC 6000 series under NOS and the CDC CYBER series under NOS.
integral	A software element is said to be integral if it contains a unique module, called the control module, which controls the invocation of other modules in the software element and in other software elements such that the entire capability of the software element may be accessed by invocation of the control module.
library	A library is a collection of software (e.g., source modules, macros, object modules) residing on an external storage device. A library has no implication of a software hierarchy associated with it.

Term	Definition
load-module	A load-module is a discrete and identifiable software unit which is output by a link editor in a form which can be loaded into memory and executed directly. A load-module contains one or more modules.
Master Data Base	The Master Data Base is a collection of data at each installation which describes aircraft, aircraft components, and other analysis components which may be analyzed; maneuvers, conditions, and operating regimes for an analysis; and failure/damage effects which might be considered. It is the intent of the System design that data which is used repeatedly in analysis at an installation will be maintained in the Master Data Base by authorized personnel at the installation.
module	A module has the following principal characteristics: (1) it performs only one function; (2) it has a unique name; (3) it constitutes one compilation or assembly; (4) it has only one point of entry. An invocable, executable module consists of no more than 100 executable source language statements. (Note that not all modules are executable; e.g., a FORTRAN block data subprogram is an example of a nonexecutable module.)
node point	A node point is a distinct location, which is part of the configuration, at which motions or

Term	Definition
node point (cont'd)	loads are specified or desired. Node points are divided into two types: dynamic node points and aerodynamic node points. A node point may be both an aerodynamic node point and a dynamic node point.
Operational Complex package	See complex.  Each subsystem is made up of packages. A package is a set of modules that together performs a related aggregate of functions of the subsystem to which it belongs. Every module of the System belongs to one and only one package. (Large packages are frequently composed of subpackages; subpackage is defined below.) Execution of the packages and subpackages of the Technology Component is controlled by the Executive Component. Packages and subpackages may be invoked either as a result of entries in the Sequence Control Table, which is constructed from user-specified data by the Executive Component during the input phase of an analysis run, or as a result of requests from a module of the Technology Component during the Processing Phase of an analysis run.
Run Data Base	The Run Data Base is a collection of data which describes the following for an analysis run: the aircraft, aircraft components, and other analysis components to be analyzed; maneuvers, conditions, and operating regimes for



Term	Definition
Run Data Base (cont'd)	the analysis; failure/damage effects to be applied during the analysis run; temporary values calculated during the analysis run for use later in the analysis run; and output values calculated during the analysis run. The Run Data Base is created for an analysis run and is destroyed at the end of the analysis run.
separable	A software element is said to be separable if it does not contain a unique module, called the control module, through which the entire capability of the software element can be accessed.
Sequence Control Table	The Sequence Control Table is a System Command Sequence in internal computer form.
set	A set (of data) is a collection of related data in a data base identified and managed as a single entity for convenience and efficiency. The Master Data Base and the Run Data Base contain sets. A set may contain other sets, arrays, and data-items.
software element	The term "software element" is used to denote, nonspecifically, a member of the System software hierarchy. The term is usually used to denote either a package or a subpackage.
subpackage	Each package may be made up of subpackages. A subpackage is a set of modules that together

Term	Definition
subpackage (cont'd)	performs a related aggregate of functions of the package to which it belongs.
subsequence	See System Command Subsequence.
subsequence set	A subsequence set is a collection of System Command Subsequences all of which perform the same major System function but which differ from each other in either the methods used to perform the common function or the level of detail at which the common function is analyzed.
subsystem	Each complex is made up of subsystems. A subsystem is a set of modules that performs a related aggregate of System functions. A subsystem is not an executable entity. Subsystems are defined to relate the software design of the System to the requirements of the Type A System Specification; to aid in presenting a logical, top-down design; and to aid in configuration management. Every module of the System belongs to one and only one subsystem. In the Operational Complex, a subsystem is characterized as being an executive subsystem or a technology subsystem. The collection of executive subsystems is called the Executive Component, and the collection of technology subsystems is called the Technology Component. The Executive Component provides the functions of user interface, run-time management, data base management, and operating system interface.

Term	Definition
subsystem (cont'd)	The Technology Component provides all the functions of helicopter and associated mathematical analyses. These two components are mutually exclusive and exhaustive; i.e., a module in the Operational Complex is in one and only one component.
Support Complex	See complex.
System	The word System (note the upper case S) is a shorthand equivalent for the Second Generation Comprehensive Helicopter Analysis System. The System, which is a software system, is to be developed by the Government to accurately predict performance, stability and control, loads and vibrations, acoustics, and aeroelastic stability for a variety of rotary-wing aircraft configurations. The System is made up of a set (collection) of modules. Because a module is a relatively small part of the System (a module consists of no more than 100 executable source language statements), names are given to particular sets of modules in the System. The names used to describe the System's hierarchy are complex, subsystem, package, and subpackage. These sets of modules are, in the terminology of set theory, subsets of the System. The System is composed of two complexes; a complex in turn is composed of subsystems; a subsystem is composed of packages; a



Term	Definition
System (cont'd)	package may be composed of sub-packages; and, finally, subpackages are composed of either other subpackages or modules. Every member of this hierarchy (complex, subsystem, package, subpackage) can be thought of as being composed of modules.
System Command Sequence	A System Command Sequence is a set of System Commands which specify the actions to be taken by the System to perform an analysis. A Particular Functional Capability is represented by a System Command Sequence.
System Command Subsequence	A System Command Subsequence is a set of System Commands which perform a single major System function. System Command Subsequences may be combined to form System Command Sequences.
target computer	A target computer is a computer at a user's installation on which the System will be installed.
Technology Component	See subsystem.